Ein Beitrag zur Beschleunigung der inter-FPGA-Datenübertragung mittels asynchroner Pulslängenauswertung

Dissertation zur Erlangung des akademischen Grades Doktor-Ingenieur der Fakultät für Informatik und Elektrotechnik der Universität Rostock

vorgelegt von Matthias Hinkfoth

Rostock, den 23. Oktober 2017

urn:nbn:de:gbv:28-diss2017-0137-3

Gutachter

Prof. Dr.-Ing. habil. Ralf Salomon
Universität Rostock
Fakultät für Informatik und Elektrotechnik
Institut für Angewandte Mikroelektronik und Datentechnik

Prof. Dr.-Ing. Michael Hübner
Ruhr-Universität Bochum
Lehrstuhl für Eingebettete Systeme der Informationstechnik

Prof. Dr. Michael Kuhn
Hochschule Darmstadt
Fachbereich Elektrotechnik und Informationstechnik

Datum der Einreichung

28. April 2017

Datum der Verteidigung

18. September 2017

Für Conny, Benjamin, Samuel und Rebekka

Inhaltsverzeichnis

1 Einleitung

Die Kommunikation mittels elektronischer Geräte hat in den letzten Jahrzehnten eine enorme gesellschaftliche Bedeutung erlangt. Die digitale Datenübertragung stellt eine wesentliche Grundlage dieser Kommunikation dar. Und obwohl sich heutzutage viele Menschen per Funk mit dem Internet verbinden, bildet die kabelgebundene Infrastruktur einen wichtigen Teil der Kommunikationsnetze. Insbesondere innerhalb von Gebäuden sind Funkverbindungen häufig durch die Bausubstanz problematisch. Darüberhinaus wird in der Hausverkabelung gern mit bestehenden Kabelinstallationen gearbeitet, um den Aufwand und die Kosten des Verlegens zu vermeiden.

Unabhängig von der Wahl des Mediums soll eine Datenübertragung immer möglichst schnell und möglichst günstig erfolgen. Das bedeutet konkret, eine hohe Datenrate bei kurzer Verarbeitungsdauer, niedrigem Energiebedarf und geringem Schaltungsaufwand zu erreichen. Ein Problem dabei stellt die Ausnutzung der zur Verfügung stehenden Bandbreite des Kabels dar. Aktuelle Bit-serielle Verfahren, wie z.B. USB 3.1, erreichen 10 GBit/s auf einer Länge von bis zu 2 m. Für längere Übertragungswege muss aufgrund der begrenzten Bandbreite ein anderes Protokoll mit reduzierter Datenrate verwendet werden. Das bedeutet, dass die Auswahl der Übertragung anhand von möglicherweise noch unbekannten Parametern wie Kabellänge und Kabelbandbreite erfolgen muss.

Sollen dennoch hohe Datenraten auf längeren Kabeln erzielt werden, bieten sich höherwertige digitale Modulationsverfahren an. Beispiele dafür sind die Pulsamplitudenmodulation bei Gigabit-Ethernet und die Quadratur-Amplituden-Modulation bei ADSL. Diese Art der Modulation führt jedoch zu einem vergleichsweise hohen Schaltungsaufwand. Die zusätzlich notwendigen analogen Schaltungsteile erfordern zusätzliche Entwicklungzeit, zusätzliche Energie im Betrieb und verursachen während der Übertragung zusätzliche Verzögerungen durch die Signalverarbeitung.

Wünschenswert ist daher eine Datenübertragung, welche die Vorteile der höherwertigen Modulation, nämlich die hohe Datenrate und die Anpassungsfähigkeit an die Bandbreite, mit den Vorteilen der Bit-seriellen Übertragung nämlich dem niedrigen Schaltungsaufwand und der niedrigen Verzögerung kombiniert. Im Idealfall zeichnet sich die Lösung darüberhinaus durch einen niedrigen Energiebedarf und eine einfache Umsetzbarkeit aus.

Diese Arbeit stellt auf Basis der Pulsweitenmodulation (PWM) ein Datenübertragungssystem vor, das die obengenannten Anforderungen erfüllt. Zu diesem Zweck stellt Kapitel 2 an Beispielen grundlegende Begriffe und Größen zum Vergleich verschiedener Datenübertragungsverfahren vor, und geht auf die Wichtigkeit der häufig vernachlässigten Latenz ein. Im Anschluss beschreibt Kapitel 3 mit Field-Programmable Gate Arrays (FPGAs) eine geeignete Plattform für die in dieser Arbeit vorgenommene prototypische Implementierung der vorgeschlagenen PWM-Datenübertragung. Das darauffolgendende Kapitel 4 stellt daher FPGA-basierte Systeme zur Zeitmessung vor, da die Bestimmung der Pulslänge eine wesentliche Anforderung der PWM-Demodulation ist. Der Fokus liegt dabei auf der Anwendung asynchroner, hochpräziser, FPGA-basierter Zeitmesser, wie sie sonst z.B. in der Teilchenphysik Verwendung finden.

Aufbauend auf den in den Grundlagen beschriebenen Technologien entwirft Kapitel 5 die Idee einer PWM-Datenübertragungsstrecke, und leitet ihre mathematischen Eigenschaften her. Kapitel 6 beschreibt eine Reihe entsprechender Implementierungsmöglichkeiten auf FPGAs, und behandelt jeweils ihre Vorzüge und Nachteile. Hauptaugenmerk liegt auf den asynchronen Implementierungen von Modulator und Demodulator. Wesentliche Aspekte des Entwurfs sind geringer Platz- und Energiebedarf sowie einfache Umsetzbarkeit auf verschiedenen FPGA-Plattformen.

Die erstmalige, prototypische Implementierung einer asynchronen PWM-Datenübertragungsstrecke stellt Kapitel 7 vor. Deren in Kapitel 8 dargestellten Ergebnisse weisen mit einer Datenrate von 70 MBit/s bereits auf das Potential der PWM-Datenübertragung hin. Aufbauend auf dieser ersten Implementierung werden in den folgenden Kapiteln Ansätze zu Verbesserungen anhand zweier Anwendungsfälle untersucht. Der erste Anwendungsfall (Kapitel 9) ist die Inter-Chip-Kommunikation über Platinengrenzen hinweg. Hierbei ist die Leistungsfähigkeit der beteiligten FPGAs begrenzend für die Datenrate. Am Beispiel einer Übertragung über einen Meter Koaxialkabel, wird eine Datenrate von einem Gigabit pro Sekunde erzielt. Mit dieser Datenrate übertrifft die PWM-Datenübertragung die maximale Bit-serielle Übertragungsgeschwindigkeit des Cyclone II FPGA um 50 %. Damit stellt das Kapitel die absolute Leistungsfähigkeit der PWM-Datenübertragung in Anbetracht der begrenzten Leistungsfähigkeit der Hardware dar.

Der zweite Anwendungsfall (Kapitel 10) untersucht die PWM-Datenübertragung unter anderem auf einem 223,2 m langem, verdrilltem, symmetrischen Kabel. Dies geschieht insbesondere im Hinblick auf die Adaptivität des Verfahrens an unbekannte Übertragungsstrecken mit niedrigen Bandbreiten. Dabei kann die PWM-Datenübertragung ihre bereits theoretisch gezeigten Vorzüge gegenüber Bit-seriellen Verfahren ausspielen, und erreicht eine Steigerung der Datenrate um mehr als den Faktor drei. Bei einer Bandbreite von 6,1 MHz können 40 MBit/s erreicht werden. Kapitel 11 ordnet die gewonnenen Resultate in den Stand der Technik

ein und nimmt eine kritische Einschätzung der Ergebnisse vor. Die Arbeit schließt mit einer Reihe von Ideen für weiterführende Arbeiten.

2 Grundlagen digitaler Datenübertragung

Daten werden von einem Sender zu einem oder mehreren Empfängern übertragen. Ein häufiger Fall ist die elektrische Verbindung eines Senders mit einem Empfänger durch ein Kabel. Eine unidirektionale Datenübertragungsstrecke ist in Abbildung 2.1 dargestellt. Sender und Empfänger werden in Schichten aufgeteilt.

Jede Schicht erfüllt eine von den anderen Schichten abgegrenzte Aufgabe. Beispielsweise sorgt der Modulator im Sender für die Umsetzung logischer Zustände in physikalische Größen. Die Aufteilung in Schichten ist besonders hilfreich bei der Weiterentwicklung existierender Übertragungssysteme. Darüberhinaus ist eine isolierte Implementierung einzelner Schichten möglich, was die Entwicklung und das Testen neuer Implementierungen stark vereinfacht.

Dieses Kapitel stellt am Beispiel der physikalischen Schicht des USB-Protokolls die grundlegenden Eigenschaften von Datenübertragungssystemen vor, soweit sie für das Verständnis der Arbeit notwendig sind. Die Schwerpunkte dieser liegen dabei auf der Modulation und auf einzelnen Aspekten der Kanalcodierung. Dies beinhaltet einerseits offensichtlich relevante Größen wie die Datenrate und die Latenz der Datenübertragung. Andererseits zählen auch Implementierungsaspekte wie Schaltungsaufwand, Energiebedarf, Entwicklungszeit, Entwicklungskosten und Fertigungsaufwand dazu.

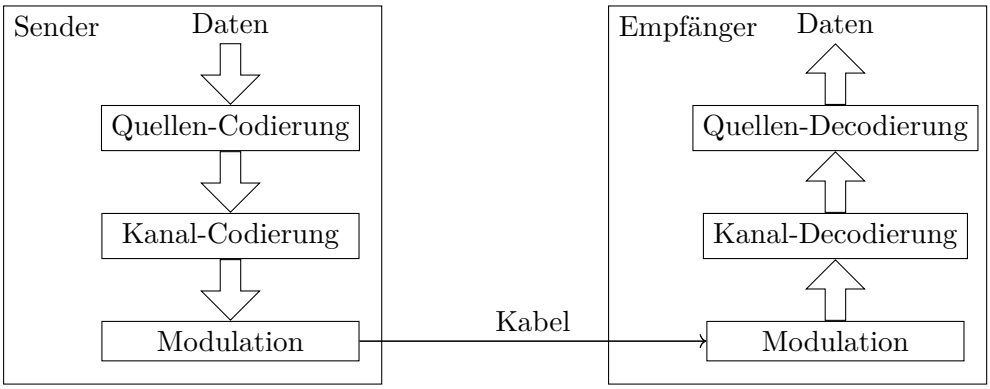

Abbildung 2.1: Blockschaltbild einer digitalen Datenübertragungsstrecke nach [Xio06, Seite 2, Abbildung 1.1].

2.1 Die physikalische Schicht von USB 3.1

Der aktuelle USB-Standard beschreibt auf über 600 Seiten vier Protokollschichten vom Physical Layer bis zum Device-/Host-Layer. Abbildung 2.2 vermittelt einen Eindruck vom Umfang des Protokoll-Stacks. Im Rahmen dieser Arbeit ist nur die als Physical Layer bezeichnete Schicht relevant. Dieser Layer enthält sowohl die Modulation als auch die Kanal-Codierung.

Die Kabelspezifikation legt für Datenverbindung von USB 3.1 ein geschirmtes, symmetrisches, verdrilltes Adernpaar je Richtung fest. Als Modulation ist eine Pulsamplitudenmodulation mit zwei differentiellen Spannungspegeln vorgegeben, also ein binäres Signal. Das differentielle Signal hat einen Spanunngshub von $V_{P-P} = 1\,\mathrm{V}$ und eine Symboldauer[1] von 100 ps. Diese Festlegungen beruhen auf den elektrischen Eigenschaften des Kabels, oder allgemeiner, des Übertragungskanals. Da der Kanal wesentlich die Möglichkeiten eines Datenübertragungssystems bestimmt, widmet sich der folgende Abschnitt den grundsätzlichen Kanaleigenschaften.

[1] im Standard als Unit Intervall bezeichnet

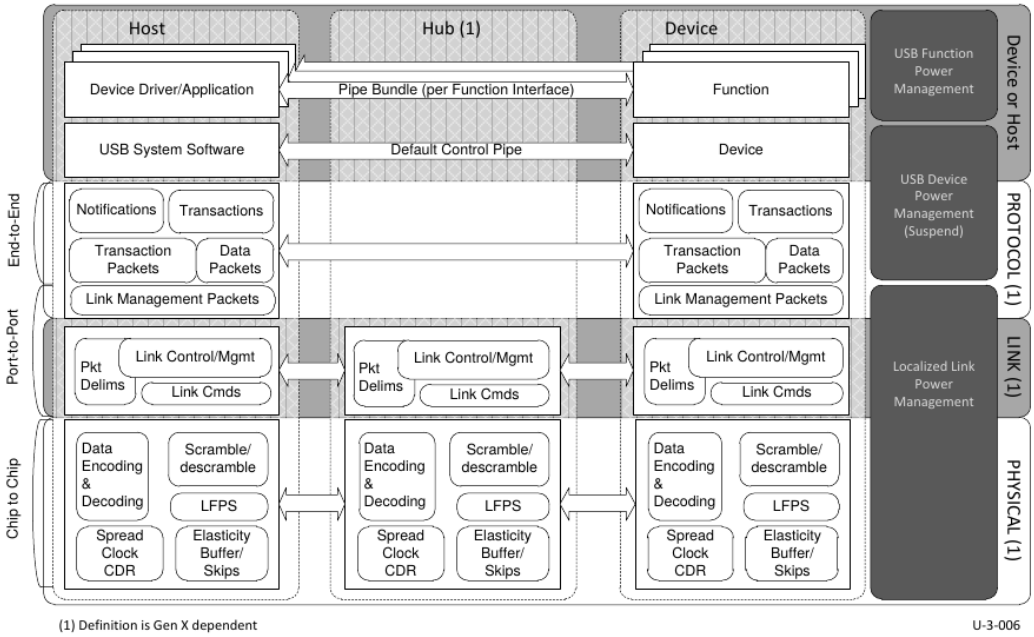

Abbildung 2.2: Eine Übersicht des Protokoll-Stacks von USB 3.1 aus [usb-spec13, Seite 3-6, Abbildung 3-4]

2.2 Kanaleigenschaften

In der Nachrichtentechnik erfolgt die Betrachtung der Eigenschaften eines Kabels im Hinblick auf die Übertragungseigenschaften anhand einer Reihe von theoretischen Größen, um daraus allgemeingültige Aussagen abzuleiten. Einige dieser Größen, wie die Bandbreite, sind auch in der Praxis relevant, können sich aber in ihrer Begriffsdefinition oder in den zugrundeliegenden Parametern unterscheiden. Darauf wird an den notwendigen Stellen eingegangen.

2.2.1 Kanalkapazität

Die Kanalkapazität ist eine theoretische Größe, die beschreibt, welche Datenmenge in einer gewissen Zeit maximal auf einem Kanal übertragen werden kann. Die Kanalkapazität hängt von zwei Größen ab: der Bandbreite des Kanals und dem Signal-Rausch-Verhältnis. Dabei wird als Rauschmodell ein additives weißes Rauschen angenommen. Dies hat eine konstante Rauschleistung über den gesamten Frequenzbereich. Als Bandbreitenbegrenzung wird ein idealer Tiefpass verwendet. Dann ergibt sich die Kanalkapazität C zu $C = B \log_2(1 + \mathrm{SNR})$. Dies ist die höchste theoretisch, und auch die maximale praktisch erreichbare Datenrate auf einem solchen Kanal. Bereits Shannon [Sha49] führt aus, dass ein zeitbegrenztes Signal nicht exakt bandbegrenzt sein kann. Zwar sind in der Praxis keine idealen Tiefpässe vorhanden, dennoch bildet ein solches Kanalmodell eine geignete Grundlage für die Abschätzung einer Reihe praktischer Größen.

2.2.2 Kanalmodell idealer Tiefpass

Dieser Abschnitt ist eine Zusammenfassung der wesentlichen Punkte aus [Rud05]: Ein idealer Tiefpass mit der Grenzfrequenz $B = f_c = \frac{\omega_c}{2\pi}$, hat eine Übertragungsfunktion

$$H(\omega) = A_0 \cdot \mathrm{rect}_{\omega_c} \cdot e^{-jwt_0}.$$

Dessen Impulsantwort lautet

$$h(t) = \frac{A_0 \cdot \omega_c}{\pi} \cdot \frac{\sin(\omega_c(t - t_0))}{\omega_c(t - t_0)}.$$

Die Fläche unter der Impulsantwort beträgt $\int_\infty^\infty h(t)dt = A(0) = A_0$. Die mittlere Breite der Impulsantwort t_i ist nun gleich der halben Breite des Hauptmaximums Δt:

$$t_i = \frac{\Delta t}{2} = \frac{\pi}{\omega_c}.$$

Eine unter Grenze für die Anstiegszeit t_r liefert die Auswertung der Sprungantwort $a(t)$ des idealen Tiefpasssystems:

$$a(t) = \frac{A(0)}{2} + \frac{A(0)}{\pi} \int_{-\omega_c(t-t_0)}^{\omega_c(t-t_0)} \frac{sin(x)}{x} dx = \frac{A(0)}{2} \left(1 + \frac{2}{\pi} Si\left(\omega_c(t-t_0)\right)\right)$$

Die Anstiegszeit t_r wird durch eine Rampenfunktion A_0/t_r im Punkt $(t_o, A(0)/2)$ angenähert. Dies entspricht dem maximalen Anstieg der Sprungantwort

$$h(t_0) = \frac{A_0 \omega_c}{\pi} = \frac{A_0}{t_r}$$

Für die Anstiegszeit t_r folgt

$$t_r = \frac{pi}{\omega_c} = \frac{1}{2f_c} = \frac{1}{2B} = t_i$$

Damit sind die minimale Anstiegszeit t_r der Sprungantwort und die mittlere Impulsbreite t_i der Impulsantwort gleich groß [.]

Da sich bei der Bit-seriellen Übertragung aus der Impulsantwort auch die minimale Pulsbreite $t_{\text{minpuls}} = t_r = t_i$ ergibt, liefert die Abschätzung $B \geq 1/2t_{\text{minpuls}}$ eine untere Schranke für die benötigte Bandbreite eines binären Signals der Pulsbreite t_{minpuls}.

2.2.3 Dämpfung

Ein Kabel hat als elektrischer Leiter einen gewissen Leitungswiderstand und eine Kapazität[2]. Sowohl der Widerstand als auch die Kapazität sind aufgrund der Geometrie linear von der Kabellänge abhängig. Daraus leiten sich die auf die Länge normierten Größen des Widerstandsbelags und des Kapazitätsbelags ab. Der ohmsche Widerstand führt dazu, dass die vom Sender in das Kabel eingespeiste Leistung den Empfänger nicht vollständig erreicht, sondern auf dem Weg dorthin in Wärme umgesetzt wird. Der Kapazitätsbelag führt zu einer frequenzabhängigen Impedanz des Kabels, und damit zu einer frequenzabhängigen Dämpfung des Signals.

In Abbildung 2.3 sind die Dämpfungswerte verschiedener Koaxialkabel mit einer Länge von 100 m dargestellt. Auf der Abszisse ist die Frequenz des übertragenden Signals logarithmisch aufgetragen. Auf der Ordinate ist die Dämpfung in dB, welches schon eine logarithmische Größe ist, nochmals logarithmisch aufgetragen. Alle Dämpfungsverläufe sind als Geraden erkennbar. Dass bedeutet, dass die Dämpfung exponentiell mit der Frequenz ansteigt.

[2] elektrische Größe, nicht die Kanalkapazität

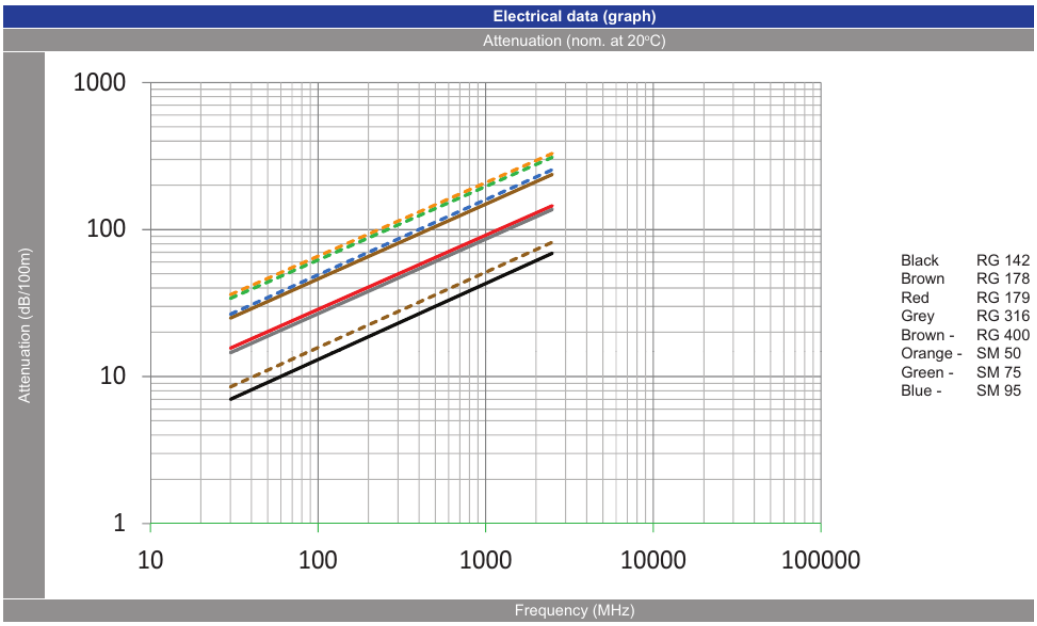

Abbildung 2.3: Die frequenzabhängige Dämpfung verschiedener Koaxialkabel, aus [RG316/U13].

Das im weiteren Verlauf der Arbeit verwendete RG316/U-Kabel ist in Abbildung 2.3 grau dargestellt. Es hat bei 100 MHz eine Dämpfung von 27 dB, bei einer Länge von 100 m [RG316/U13].

$$27\,\mathrm{dB} = 10 \cdot \log_{10}(P_{\mathrm{ein}}/P_{\mathrm{aus}}) = 20 \cdot \log_{10}(U_{\mathrm{ein}}/U_{\mathrm{aus}})$$

durch 20 dividieren

$$\log_{10}(U_{ein}/U_{aus}) = 1{,}35$$

und das Ergebnis als Exponent zur Basis 10 rechnen

$$U_{ein}/U_{aus} = 10^{1{,}35} = 13{,}6\ldots$$

Das bedeutet, dass ein 100 MHz Sinus der mit der Amplitude von 1 V in ein 100 m langes RG316/U-Kabel eingespeist wird, am Ausgang noch die Amplitude $U_{aus} = U_{ein}/13{,}6 = 1\,\mathrm{V}/13{,}6 \approx 0{,}07\,\mathrm{V}$ hat. Wenn dieser verringerte Spannungspegel, vom Empfänger nicht mehr erkannt wird, kann bei dieser Signalfrequenz keine Datenübertragung stattfinden.

Im Rahmen der binären Übertragung wirken sich die elekrischen Eigenschaften des Kabels nicht nur auf die Amplitude, sondern auch auf die Signalform der Pulse aus. Eine detaillierte Beschreibung erfolgt in Abschnitt 2.3.

2.2.4 Das Signal-Rausch-Verhältnis

Eine weitaus schwerer zu bestimmende Größe, die das Übertragungsverhalten des Kanals beeinflusst, ist das Rauschen. Als gängigstes Rauschmodell wird das additive weiße Rauschen verwendet, da es den ungünstigsten Fall darstellt, und damit als Abschätzung für alle anderen Rauscharten verwendet werden kann [Sha49].

Durch das dem Nutzsignal überlagerte Rauschen, variiert die Amplitude des Signals in gewissen Grenzen zufällig. Im Falle einer Signalflanke ist damit jedoch der Zeitpunkt des Überschreitens der Reaktionsschwelle des Empfängers nicht exakt vorhersagbar. Daher wirkt sich die Amplitudenunsicherheit aufgrund des Rauschens auch als Zeitunsicherheit aus. Dieses Rauschen im Zeitbereich wird auch als Jitter (englisch für Zittern) bezeichnet.

Die genaue Quantifizierung des Rauschens ist schwierig bis unmöglich. Daher nutzt eine theoretische Beschreibung des Kanals die mittlere Rauschleistung N im Verhältnis zur mittleren Sendeleistung P das gemittelte Leistungsdichtespektrum. Das Signal-Rausch-Verhältnis P/N dient als Bezugsgröße die unabhängig von der Sendeleistung ist. In der Praxis ist dieses Verhältnis innerhalb der physikalischen Grenzen zu einem gewissen Maße beeinflussbar, beispielsweise durch höhere Sendeleistungen P bei gleicher Empfängerempfindlichkeit.

Eine präzise Abschätzungen der Rauschleistung fällt grundsätzlich schwer, da die Rauschursachen selten genau genug bestimmt werden können. Daher sind praktische Versuche zum Einfluss des Rauschens auf eine konkrete Übertragungsstrecke unabdingbar.

2.2.5 Einfluss der Kabellänge

Viele signalbeeinflussende Größen sind von der Länge des Kabels abhängig. Darunter ist auch die Dämpfung. Mit zunehmender Kabellänge verringert sich der Anteil der Signalenergie, der den Empfänger erreicht. Darüberhinaus steigt mit der Länge auch die Angriffsfläche für Umwelteinflüsse. Insgesamt haben eine sinkende Signalamplitude und ein steigendes Rauschen somit ein kleineres Signal-Rausch-Verhältnis zu Folge. Durch den Kapazitätsbelag des Kabels sinkt die Bandbreite ebenfalls mit zunehmender Kabellänge. Beide Effekte zusammen sorgen dafür, dass auf kurzen Leitungen höhere Datenraten erreicht werden, als auf längeren.

2.2.6 Signallaufzeit

Die Signallaufzeit $t = s/v$ ist der Quotient aus Übertragungsentfernung s und Signalgeschwindigkeit v. Elektrische Signale erreichen auf Koaxialkabeln etwa 2/3 der Lichtgeschwindigkeit $v_{\text{koax}} \approx 2/3c \approx 2 \cdot 10^8\,\text{m}\,\text{s}^{-1}$, und auf Twisted-Pair-Kabeln mindestens $v_{\text{twist}} = 1\,\text{m}/5{,}7\,\text{ns} \approx 1{,}75 \cdot 10^8\,\text{m}\,\text{s}^{-1}$ (siehe [Siemon97]). Je

nach Länge des Kabels kann die Signallaufzeit eine signifikante Größe für die Datenübertragung sein.

2.3 Signalform der USB 3.1-Datenübertragung

Die Pulsdauer von 100 Picosekunden erfordert für ideal bandbreitenbegrenzte Pulse eine Bandbreite von 5 GHz. Der USB-Standard [usb-loss15, Seite 10] legt eine Gesamtdämpfung von Sender zu Empfänger von 23 dB bei 5 GHz fest. Davon entfallen 6 dB auf das Kabel. Die maximale Kabellänge liegt daher, abhängig vom Kabeltyp, zwischen einem Meter und zwei Metern (siehe auch die Dämpfungsangaben [usb-typec16, Seite 74, Tabelle 3-21]).

Signalform bei Bit-serieller Übertragung

Sämtliche elektrische Eigenschaften eines Kabels, wie Widerstandsbelag, Kapazitätsbelag und Induktivitätsbelag beeinflussen die erreichbare Datenrate. Diese Größen sind jedoch Schwankungen, z.B. abhängig vom Biegeradius, unterworfen. Ein praktisch implementiertes Übertragungssystem muss daher eine Menge bekannter und unbekannter Einflüsse berücksichtigen. Anhand der Signalform können alle Einflussgrößen zusammen betrachtet werden.

Ein übertragenes Signal lässt sich wie in Abbildung 2.4 darstellen. Jedes Bit hat eine definierte Dauer. Das Signal setzt sich aus den aufeinanderfolgenden Symbolen zusammen (a). Das reale Signal (b) hat aufgrund der Leitungswiderstände nur endlich steile Flanken. Aufgrund der Anstiegszeit der Flanken lässt sich die

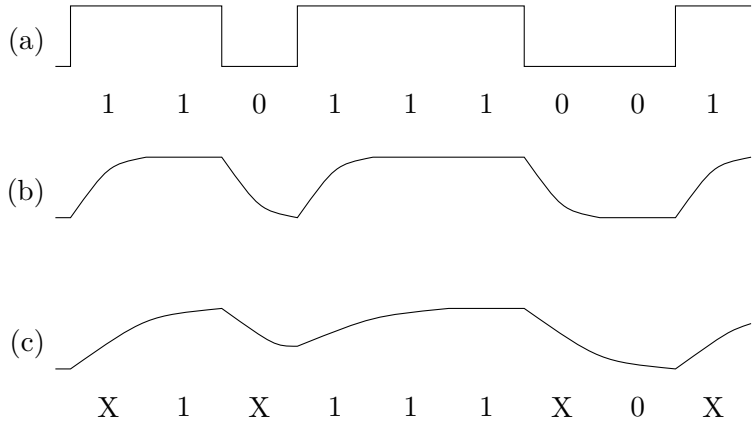

Abbildung 2.4: Bit-serielles Signal mit (a) idealisierter Signalform, (b) realer Signalform mit ausreichend steilen Flanken und (c) realer Signalform mit zu geringer Flankensteilheit.

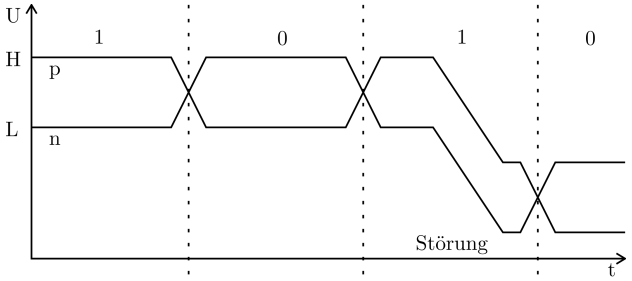

Abbildung 2.5: Beispielhafte Spannungen bei differentiellen Signalen. Der Bit-Wert ist Eins, wenn p größer als n ist, und Null, wenn n größer als p ist. Die Störung verschiebt die Potentiale beider Leitungen nach unten, jedoch bleibt der Bit-Wert 1 erhalten.

Symboldauer nicht beliebig kurz wählen. Unterschreitet die Symboldauer diese Anstiegszeit, so können einige Symbole nicht mehr erkannt werden (c). Diese Eigenschaft ist für die Datenübertragung von grundlegender Bedeutung.

Differentielle Signalpegel

Ein wesentliches Problem der Signalübertragung stellen störende Spannungseinflüsse aus der Umgebung dar. Die differentielle Übertragung beruht auf der Differenz zweier Potentiale, die auf benachbarten elektrischen Leitern übertragen werden. Der Empfänger vergleicht den Signalpegel nicht mit einem festen Bezugspotential, sondern mit dem vom Sender übertragenen variablen Potential. Dadurch können Gleichtaktstörungen das Signal nicht mehr beeinflussen. Ebenso wirken sich externe Rauschsignale ähnlich auf beide Leitungen aus. Abbildung 2.5 stellt den Signalverlauf auf einem differentiellen Aderpaar dar. Die Signale sind mit p und n bezeichnet. Der logische Wert wird aus der Differenz der beiden Signalpegel bestimmt. Abgesehen von der elektrischen Differenzbildung verhält sich das Signal einer differentiellen Übertragung ebenso wie ein „normales“ Spannungssignal. Die folgenden Ausführungen gelten daher für ein single-ended Signal ebenso, wie für das Differenzsignal einer symmetrischen Übertragung.

Elektrische Tricks

Es kommen zwei elektrische Tricks zum Einsatz, die die Flankensteilheit vergrößern: De-Emphasis im Sender und ein Equalizer im Empfänger. Die De-Emphasis besteht aus einem 3-stufigen FIR-Filter, das eine Hochpassfunktion realisiert. Der Equalizer im Empfänger ist ein einstellbarer Verstärker, der in Kombination mit einem einstellbaren Filter ebenfalls Hochpasscharakter aufweist. Mit Hilfe dieser Maßnahmen, erreicht USB 3.1 die nötige Flankensteilheit für 100 ps-Pulse.

2.4 Kanalkodierung von USB 3.1

Die Kanalcodierung erfolgt auf Basis von 128 Daten-Bit, also 16 Byte. Diese werden mit dem Ausgang eines LFSR per XOR verknüpft. Den verwürfelten Bit wird ein 4 Bit langer Header vorangestellt. Ein Datenrahmen[3] hat damit eine Länge von 132 Bit. Der USB-Standard bezeichnet dieses Vorgehen als 128b/132b-Encoding.

2.4.1 Quellensynchrone serielle Übertragung

Schnelle serielle Verbindungen nutzen ein einziges Kabel als Verbindung, und übertragen auf diesem Kabel sowohl die Daten, als auch die Taktinformation. Die Taktinformation wird geschickt im Signal codiert, wodurch im Vergleich zur separaten Taktleitung kein Zeitversatz zwischen Daten und Takt entstehen kann. Der Empfänger kann aus dem Empfangssignal den Sendetakt rekonstruieren. Dies geschieht aufgrund der Codierung synchron zu den empfangenen Daten. Daher heißt diese Betriebsart eines Empfängers auch quellensynchron. Ein solches selbstgetaktetes Signal kann der Empfänger nahezu unabhängig vom Zeitverhalten des Kabels decodieren. Dadurch sind Bit-Dauern von 100 ps bei USB 3.1 [usb-spec13, Kapitel 6.7.1, Seite 6-30, Tabelle 6-17] möglich. Die Datenraten liegen bei 10 GBit/s pro Aderpaar. Bei geringeren Frequenzen, kann ein selbstgetaktetes Signal mehrere Kilometer weit übertragen werden (siehe Abbildung 2.6).

[3] Der Rahmen wird im USB-Standard als Block bezeichnet.

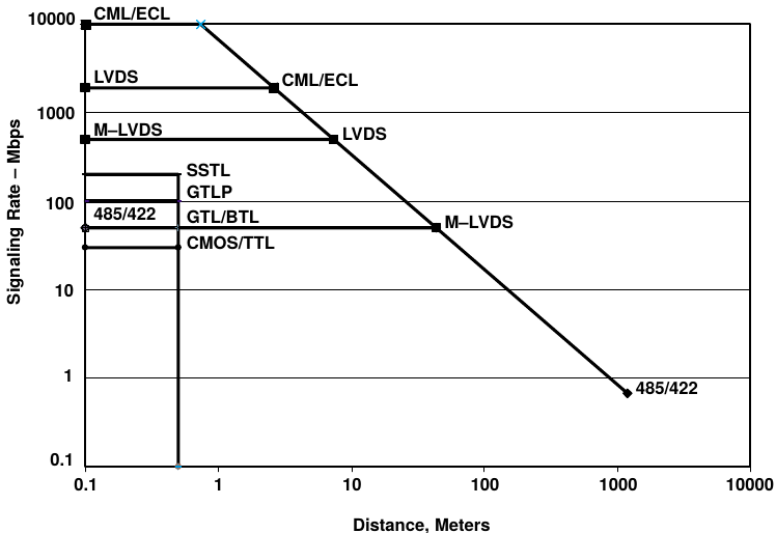

Abbildung 2.6: Übersicht der Datenraten verschiedener Bit-serieller Verfahren über die Entfernung (aus [TI02, Seite 1-1, Abbildung 1-1]).

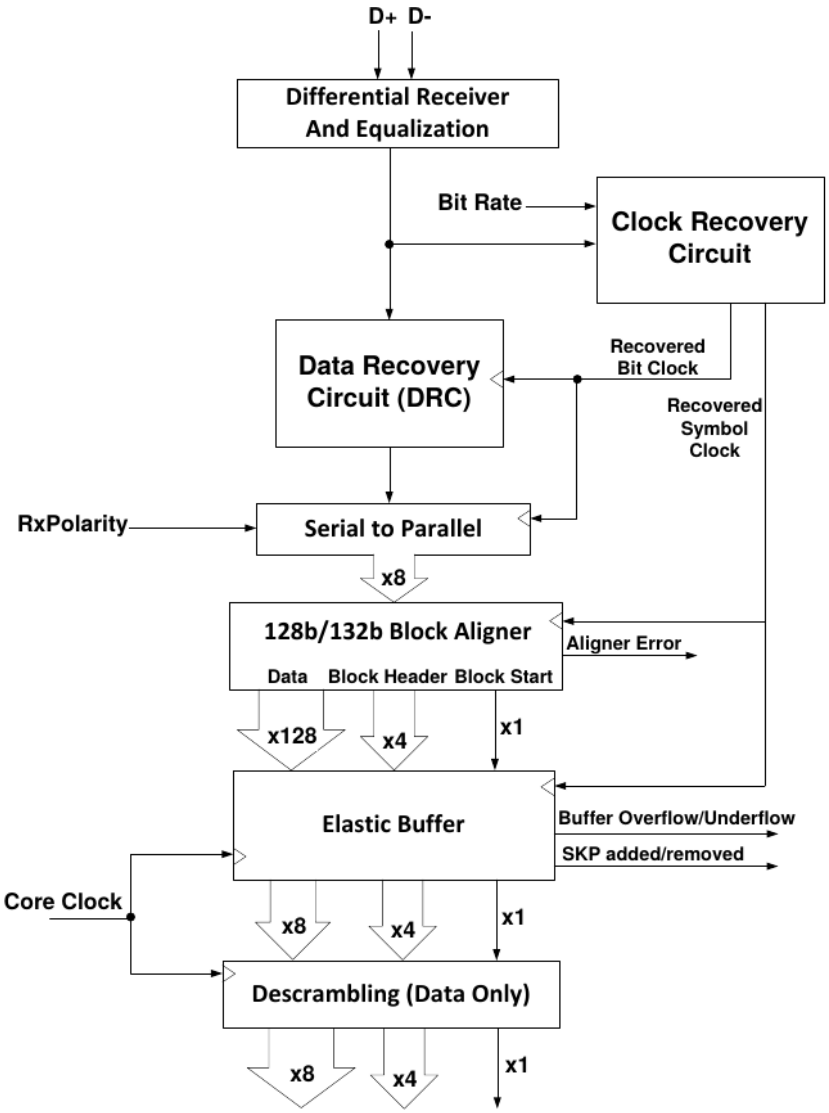

Abbildung 2.7: Taktverteilung im quellensynchronen Empfänger bei USB 3.1 (aus [usb-spec13, Kapitel 6.2, Seite 6-4, Abbildung 6-4]).

2.4.2 Taktrückgewinnung

In Abbildung 2.7 ist die Struktur der Empfangsschaltung bei USB 3.1 dargestellt. Sender und Empfänger verfügen über nominell gleichschnelle Taktquellen. Allerdings unterscheiden sich die Taktfrequenzen beispielsweise aufgrund verschiedener Umgebungstemperaturen. Daher wird eine Taktrückgewinnung des Sendertaktes im Empfänger benötigt. Dabei werden die Signalsymbole so ausgewählt (codiert), dass eine im Empfänger befindliche PLL („Clock Recovery Circuit“) synchron zu den Datensymbolen schwingt. Ein Teil der Empfängerschaltung arbeitet dann mit

der Taktfrequenz des Senders („Recovered Bit Clock“). Die Synchronisierung der Daten auf den eigentlichen Empfängertakt erfolgt im „Elastic Buffer“. Dort wird auch der durch die Frequenzabweichung entstehende, sich aufintegrierende Fehler mittels Synchronisierungssymbolen („SKP“) berücksichtigt. Dabei handelt es sich um Symbole, die vom Empfänger verworfen, oder verdoppelt werden können, und nicht der Datenübertragung dienen.

2.4.3 Eigenschaften

Mit differentiellen Signalpegeln sind quellensynchrone serielle Datenübertragungen relativ störunempfindlich. Durch die implizite Taktübertragung sind Symbolraten bis zur Bandbreitenbegrenzung möglich, da kein Jitter bezüglich eines separat übertragenen Taktsignales auftreten kann. USB 3.1 erreicht die Datenrate von 10 GBit/s bis zu einer Kabellänge von etwa drei Meter. Ähnliche Verfahren wie PCI-Express[4] werden mit noch höheren Taktfrequenzen (bis zu 16 GHz) auf kürzeren Entfernungen betrieben.

2.5 Bewertung Bit-serieller Übertragungssysteme

Die hier vorgestellte Form der quellensynchronen seriellen Übertragung ist mit mehreren Gigabit pro Sekunde um Größenordnungen schneller als klassische serielle Übertragungen. Nachteile sind der hohe Bandbreitenbedarf und die damit einhergehende Beschränkung auf kurze Entfernungen im einstelligen Meterbereich. Desweiteren ist Hardware erforderlich, die mit Taktfrequenzen im Bereich von 10 GHz arbeitet, was mittelfristig nur mit spezifisch gefertigten Schaltkreisen (ASICs) möglich ist.

2.6 Allgemeine Bewertungskriterien für Datenübertragungssysteme

Im Folgenden werden Kriterien entwickelt, anhand derer die bisher beschriebenen Bit-seriellen Übertragungsverfahren mit anderen Verfahren verglichen werden können. Die gebräuchlichste Metrik für Datenübertragungen ist die Datenrate. Implizit ist damit in der Regel gemeint, dass die Daten möglichst schnell übertragen werden. Interessanterweise ist die Datenrate nicht die einzige Größe, welche die Dauer der Datenübertragung bestimmt.

[4] Für PCI-Express ist keine frei zugängliche Spezifikation verfügbar.

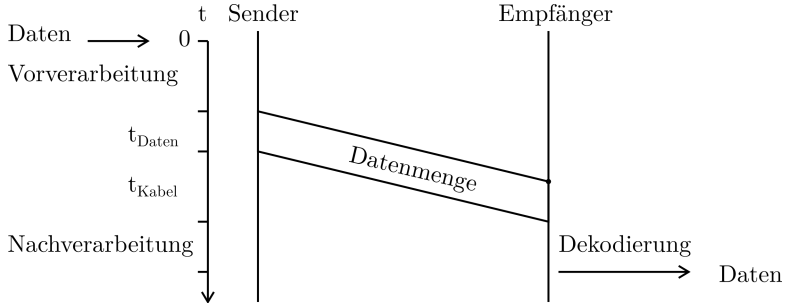

Abbildung 2.8: Die Dauer einer Datenübertragung setzt sich aus mehreren, voneinander unabhängigen Zeiten zusammen.

2.6.1 Die Dauer einer Datenübertragung

Die Dauer einer Datenübertragung ist in Abbildung 2.8 dargestellt. Sie setzt sich aus vier wesentlichen Teilen zusammen. Der erste Teil ist die Vorverarbeitungsdauer im Sender. Bei USB 3.1 zählt dazu die parallel-seriell-Wandlung und das Verwürfeln innerhalb der Kanalcodierung. Der zweite Teil ist die Rahmendauer. Sie berechnet sich aus

$$t_{\text{Daten}} = \frac{\text{Datenmenge}}{\text{Datenrate}}$$

und ist die einzige Größe, die wirklich von der Datenrate abhängt. Für einen USB 3.1-Datenrahmen beträgt diese Dauer $132\,\text{Bit} \cdot 100\,\text{ps/Bit} = 1{,}32\,\text{ns}$. Der dritte Teil ist die Signallaufzeit t_{Kabel}. Sie berechnet sich aus

$$t_{\text{Kabel}} = \frac{\text{Kabellänge}}{\text{Ausbreitungsgeschwindigkeit}}.$$

Ein 1 m langes USB 3.1-Kabel hat bei 2/3 der Lichtgeschwindigkeit eine Signallaufzeit von $t_{\text{Kabel}} = 1\,\text{m}/(2 \cdot 10^8\,\text{m}\,\text{s}^{-1}) = 5\,\text{ns}$. Der vierte und letzte Teil ist die Verarbeitungsdauer im Empfänger. Bei USB 3.1 beinhaltet dies die seriell-parallel-Wandlung, das Einsychronisieren auf den Empfängertakt und das Entwürfeln der verwürfelten Daten.

Zusätzliche Verarbeitungsschritte auf höheren Protokollschichten können die Verarbeitungsdauer signifikant erhöhen. Insbesondere Fehlerkorrektur-Codes erfordern bei blockbasierter Arbeitsweise die Übertragung ganzer Blöcke für die Auswertung einzelner Bit. Die höheren Protokollschichten werden jedoch im Rahmen dieser Arbeit nicht betrachtet. Doch auch mit den hier betrachteten niedrigen Schichten, ergibt sich für die Dauer einer Datenübertragung ein Zusammenhang von vielen Parametern. Nicht alle Parameter sind dabei von vornherein bekannt, sodass bei der Entwicklung einer Datenübertragung häufig Annahmen für die Paramterwerte getroffen werden müssen. Darüberhinaus sind für eine Hardware-Implementierung eine Reihe von physikalischen Randbedingungen zu beachten, die in der Regel den Entwurfsraum deutlich einschränken. Beispielsweise ist die

verfügbare elektrische Leistung begrenzt. Die einzelnen Randbedingungen und Parameter werden nachfolgend detailliert erläutert.

2.6.2 Datenrate

Die Datenrate R bezeichnet die gemittelte Datenmenge, die pro Zeiteinheit übertragen wird. Die maximal erreichbare Datenrate R lässt sich aus der Symboldauer $T_{\text{Symbol}} = 100\,\text{ps/Symbol}$ und dem Datengehalt von einem Bit je Symbol bestimmen. Sie liegt bei

$$R = \frac{b}{T_{\text{Symbol}}} = \frac{1\,\text{Bit/Symbol}}{100\,\text{ps/Symbol}} = 10\,\text{GBit/s}.$$

Durch den Header reduziert sich die Nutzdatenrate R_{Rahmen} auf

$$R_{\text{Rahmen}} = \frac{128}{132} \cdot R \approx 9{,}697\,\text{GBit/s}$$

2.6.3 Latenz

Die Signallaufzeit und die Verarbeitungsdauer werden unter dem Begriff Latenz zusammengefasst. Die Latenz beschreibt die Zeit, die vom Anstoßen der Übertragung im Sender vergeht, bis am Empfänger das erste Bit zur Verfügung steht. Die Signalverarbeitungsdauer ist von der konkreten Implementierung von Sender und Empfänger abhängig. Die Verarbeitungsdauer kann bei kurzen Übertragungsstrecken von wenigen Metern die Latenz dominieren.

2.6.4 Spektrale Effizienz

Anhand des in Unterabschnitt 2.2.2 vorgestellten Kanalmodells des idealen Tiefpasses ergibt sich die minimale Symboldauter T_{Symbol} zu $\min(T_{\text{Symbol}}) = 1/2B$. Die bisher vorgestellten Übertragungsverfahren unterscheiden genau zwei verschiedene Symbole, beispielsweise zwei Spannungspegel. Damit kann ein Bit pro Symbol übertragen werden. Daher begrenzt die Bandbreite B die Datenrate auf $R = b/T_{\text{Symbol}} = b \cdot 2B = 1\,\text{Bit/Symbol} \cdot 2B = 2B\,\text{Bit}$. Das Verhältnis von Datenrate R zur benötigten Bandbreite B wird als spektrale Effizienz $E = R/B$ bezeichnet. Die gebräuchliche Einheit ist Bit/s/Hz, was der Datenrate je Hertz Bandbreite entspricht. Die Bit-serielle Übertragung erreicht höchstens eine spektrale Effizienz von $E = 2\,\text{Bit/s/Hz}$. Eine weitere Erhöhung ist nur durch das Zusammenfassen mehrerer Bit in einem Symbol möglich. Dies erfordert ein höherwertiges Modulationsverfahren.

2.6.5 Gesamtdauer

Die Gesamtdauer der Datenübertragung bestimmt, wie schnell der Empfänger auf die in den Daten steckende Information reagieren kann. Sie ist damit für Datenübertragungen in Steuerungsanlagen von entscheidender Bedeutung. Um eine geringe Übertragungsdauer für eine gegebene Datenmenge zu erreichen, muss demzufolge die Datenrate möglichst hoch und die Latenz möglichst klein sein. Sobald die Leitung eine gewisse Länge erreicht, wird jedoch die Signallaufzeit der Leitung dominierend für die Latenz.

2.6.6 Energiebedarf

Der Energiebedarf einer Übertragungsstrecke ist ein nicht zu vernachlässigender Punkt. Das Erreichen hoher Datenraten erfordert nicht selten hohe Taktfrequenzen bis in den zweistelligen Gigahertzbereich. Daher müssen schon beim Entwurf die Möglichkeiten der Wärmeabfuhr von Sender und Empfänger bedacht werden.

2.6.7 Hardware-Aufwand

Sender und Empfänger sind heutzutage jeweils als integrierter Schaltkreis (auf englisch: integrated circuit; kurz IC) implementiert. Ein entscheidender Faktor für die Schaltkreisentwicklung ist die Frage, ob das Übertragungsverfahren binär arbeitet. Solche Verfahren mit nur zwei verschiedenen Spannungspegeln können mit Digital-ICs umgesetzt werden. Im Gegensatz dazu erfordern Verfahren, die auf der Unterscheidung von mehr als zwei Spannungspegeln basieren, analoge Komponenten zur Signalerzeugung und Erkennung. Die Schaltkreise müssen dann als Mixed-Signal-ICs entworfen werden. Sie enthalten sowohl analoge als auch digitale Schaltkreisteile. Für den Entwicklungsprozess einer Schaltung bedeutet das, dass die entsprechenden Kenntnisse im analogen Schaltkreisentwurf benötigt werden. Darüberhinaus sind Mixed-Signal-ICs teurer in der Entwicklung, da mehr Siliziumversionen getestet und korrigiert werden müssen als bei reinen Digital-ICs. Somit ergeben sich eine längere Entwicklungsdauer und höhere Entwicklungskosten.

2.7 Zielkonflikt: Datenrate und Hardware-Aufwand

Für ein Datenübertragungssystem sind eine hohe Datenrate, geringer Energiebedarf und geringer Hardwareaufwand wünschenwert. Das bedeutet im Einzelnen: kleine Hardware, geringe Taktfrequenz, geringe Betriebsspannung, geringer Spannungspegel der Leitung, wenige Verarbeitungsstufen, Verzicht auf analoge Schaltungsteile. Da sich diese Ziele widersprechen, muss für eine konkrete Datenübertragungsstrecke ein Kompromiss aus den Anforderungen gefunden werden. Diese

Abwägung wird häufig dadurch erschwert, dass nicht alle Parameter hinreichend bekannt sind. Daher ist es von Vorteil, wenn das System an die Umgebungsparameter angepasst werden kann.

2.8 Höherwertige Basisbandmodulation

Die Bit-serielle Übertragung kann unter idealen Bedingungen eine spektrale Effizienz von maximal 2 Bit/s/Hz erreichen. Eine Unterscheidung von vier Spannungspegeln ermöglicht beispielsweise die Übertragung von zwei Bit pro Symbol. Das entspricht einer Verdoppelung der spektralen Effizienz gegenüber der Bit-seriellen Übertragung. Dieser Ansatz wird als Pulsamplitudenmodulation (PAM) bezeichnet. Anhand der PAM werden im Folgenden die wesentlichen Eigenschaften von höherwertigen Basisbandmodulationsverfahren vorgestellt, und im Vergleich zur Bit-seriellen Übertragung bewertet.

2.8.1 Pulsamplitudenmodulation am Beispiel des Gigabit-Ethernet

Die Gigabit-Ethernet-Variante 1000BASE-T unterscheidet bei dem elektrischen Signal auf der Leitung fünf Spannungspegel. Daher heißt dieses Verfahren auch 5-PAM. Die Unterscheidung von mehr als zwei Spannungspegeln erfordert im allgemeinen zusätzliche Hardware für das Erzeugen und Erkennen der Symbole. Einerseits sind meist analoge Komponenten für die Signalerfassung und -generierung nötig, und andererseits erfolgt die Vor- und Nachverarbeitung der Signale in relativ aufwendigen digitalen Schaltungsteilen. Insbesondere analoge Verstärker mit großem Dynamikbereich tragen erheblich zum Energiebedarf der Schaltung bei [Agilent01, Seite 27].

Der wesentliche Vorteil einer höherwertigen Modulation ist, dass im Rahmen der Kanalkapazität durch alleinigen Austausch der Endgeräte gegen komplexere Hardware höhere Datenraten mit der bestehenden Leitungsinfrastruktur erreicht werden. Dieser Vorteil wird in der Regel durch einen höheren Chip-Flächenbedarf, höheren Entwicklungsaufwand, höhere Latenz und höheren Energiebedarf erkauft. Für Gigabit-Ethernet bedeutet das konkret: Die spektrale Effizienz ist besser als bei Bit-serieller Übertragung. Dadurch kann die hohe Datenrate erreicht werden. Allerdings ist zusätzliche Hardware für die Amplitudenunterscheidung nötig. Demzufolge steigt der Schaltungsaufwand.

2.8.2 Pulsweitenmodulation

Bei heute üblichen Modulationsverfahren werden ausgeklügelte analoge Schaltkreise eingesetzt. Darüberhinaus existieren jedoch einige, in der Datenübertra-

gung eher unübliche Modulationsverfahren, die als digitale Schaltung implementiert werden können. Eine solche Modulation ist die Pulsweitenmodulation.

Die Pulsweitenmodulation (PWM) bezeichnet die Variation einer Pulsdauer in Abhängigkeit der Eingangsdaten. In Abbildung 2.9 sind acht mögliche Pulse dargestellt. Ein Puls beginnt mit einer steigenden Flanke, und endet mit einer fallenden Flanke. Die Pulsdauer, also die Zeit zwischen steigender und fallender Flanke, unterscheidet sich jeweils um die „Pulsauflösung“ t_{abstand}. Die Symbollänge ergibt sich als Produkt von „Pulsauflösung“ t_{abstand} und der Anzahl verschiedener Symbole n_{Symbol}. Bei der isochronen PWM haben alle Symbole die selbe Länge [Xio06, Seite 84].

Die in Abbildung 2.9 dargestellten Symbole 0 und 7 sind nach obiger Festlegung eigentlich gar keine richtigen Pulse, denn ihnen fehlen die Flanken. Ob diese beiden Symbole für die Datenübertragung verwendet werden können, hängt vom Aufbau des Empfängers ab. Benötigt der Empfänger in jedem Puls eine Flanke, so dürfen nur „echte“ Pulse verwendet werden. Die ausschließliche Nutzung echter Pulse kann darüberhinaus sinnvoll sein, wenn das Übertragungsprotokoll die konstanten Spannungspegel für andere Zwecke vorsieht.

Um Daten von einem Sender zu einem Empfänger übertragen zu können, müssen beide Geräte die übertragenen Signale gleich interpretieren. Im Falle der Pulsweitenmodulation bedeutet das, dass Sender und Empfänger einem Puls den selben Datenwert zuordnen. Dabei spielt die Amplitude des Pulses keine Rolle. Statt dessen sind die Daten in der Länge des Pulses codiert.

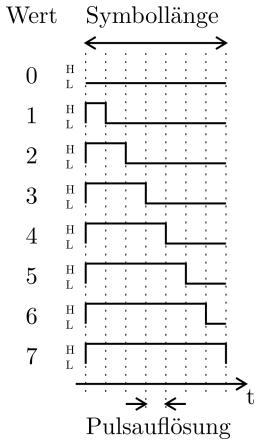

Abbildung 2.9: Bei der Pulsweitenmodulation haben verschiedene Symbole unterschiedliche Pulslängen.

Anwendung der PWM

Am bekanntesten sind die Anwendungen der PWM im Bereich von Motorsteuerungen und Lichtdimmern, wo der Gleichanteil von Strömen und Spannungen entsprechend dem Tastgrad geregelt wird. Doch auch zur Datenübertragung wird die Pulsweitenmodulation schon relative lange (seit 1980 [Sch80]) benutzt. Eine aktuelle Anwendung ist die Übertragung von Öltemperatur und Ölfüllstand bei Automotoren über ein einziges Kabel. In der Nachrichtentechnik spielt die PWM, auch als Pulsdauermodulation ([Kam08, Seite 203]) bezeichnet, eine untergeordnete Rolle.

Aufbau einer PWM-Datenübertragung

Für die Umwandlung der Datenbits in eine Pulslänge existieren verschiedene Möglichkeiten. Ein Ansatz in digitalen Schaltungen ist das Abzählen der Pulsdauer mit einem Zähler, wie es in Abbildung 2.10 dargestellt ist. Damit sind Pulslängen in ganzzahligen Vielfachen der Taktperiode möglich. Der Ausgangspuls wird durch einen Zähler erzeugt. Im Ruhezustand ist der Ausgang '0'. Sobald der Zähler zu zählen beginnt, ist der Ausgang '1'. Der Ausgang bleibt '1' solange der Zähler den Datenwert noch nicht erreicht hat. Dann stoppt der Zähler, und der Ausgang ist wieder '0'. Die Daten bestimmen also die Dauer des Zählvorgangs, und dadurch den Zählerendwert. Damit ist jedem Eingangsdatenwert genau ein Puls eindeutig zugeordnet.

Die Datenrate der PWM berechnet sich, wie bei den anderen Modulationsverfahren auch, durch die Division der Anzahl Bit $b = \log_2(n_{\text{Symbol}})$ durch die Symbollänge T_{Symbol}. Der wählbare Parameter ist die Anzahl unterscheidbarer Symbole, also die Größe des Alphabets. Abhängig von minimaler Pulsbreite t_{minpuls} und Zeitauflösung t_{abstand} ergibt sich eine für die maximale Datenrate R optimale Symbolanzahl n_{Symbol}. Die mathematische Herleitung erfolgt in Abschnitt 5.3.

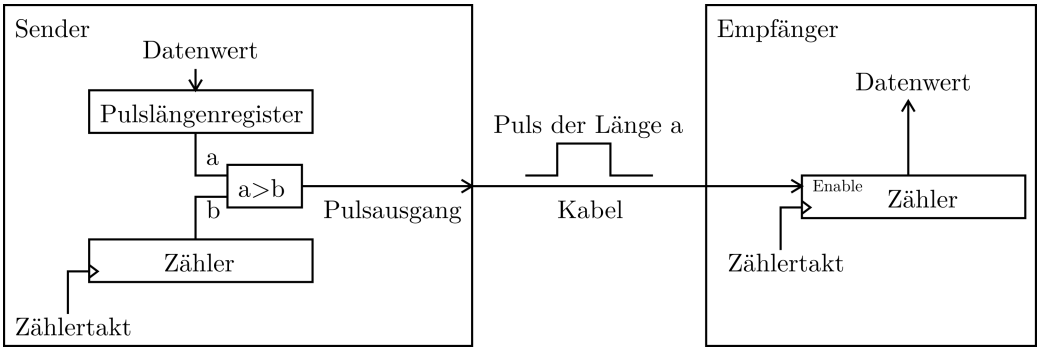

Abbildung 2.10: PWM-Datenübertragung mit synchronen Zählern.

Bewertung der PWM-Datenübertragung

Die Pulsweitenmodulation bietet gegenüber anderen höherwertigen Modulationsformen einen enormen Vorteil: Sie ist vollständig digital durchführbar. Die Demodulation kann, wie im Patent [Sch80] dargestellt, als Zeitmessung mit einem synchronen Zähler durchgeführt werden. Durch den einfachen, synchronen Aufbau ergeben sich zwei Vorteile: Die Schaltung ist temperaturunabhängig und robust gegenüber Umgebungseinflüssen. Ein wesentlicher Nachteil ist die niedrige Datenrate durch die schlechte Ausnutzung der Taktfrequenz von Sender und Empfänger. Für die Übertragung mehrerer Bit pro Symbol werden exponentiell viele Takte pro Symbol benötigt. Bei drei Bit pro Symbol sind das bereits acht Takte. Positiv ist hervorzuheben, dass bei unverändertem Übertragungskanal eine bessere Zeitauflösung der Endgeräte zu einer höheren Datenrate führt.

2.9 Vergleich der Datenübertragungsverfahren

Auf Strecken unter 10 cm erreichen parallele Datenübertragungen nach wie vor die höchsten Datenraten und niedrigsten Latenzen. Ein Beispiel dafür ist die Inter-Chip-Kommunikation über Leiterplatten, wie die DRAM-Anbindung an eine CPU mit 128 Datenleitungen. Bei parallelen Übertragungen ist die Leitungsführung sehr aufwändig, da bei hohen Datenraten alle Leitungen sehr präzise die gleiche Signallaufzeit aufweisen müssen [FL12]. Aufgrund von Umgebungseinflüssen ist dies nur bis zu einer gewissen Länge möglich. Entfernungen die größer als 10 cm sind, werden in der Regel mit seriellen Verfahren, wie z.B. USB überbrückt. Bei der Bit-seriellen Datenübertragung ist die Bandbreite der Leitung begrenzend für die Datenrate. Dies führt bei Kabellängen im Bereich von 100 m, wie sie im LAN üblich sind, dazu, dass das Potential der Sender- und Empfängerhardware nicht ausgeschöpft wird, da deren hohe mögliche Taktrate nicht genutzt werden kann. Im Gegenzug beträgt bei kürzerer Kabellänge im einstelligen Meterbereich, wie bei USB 3.1, die genutzte Kabelbandbreite mehrere Gigahertz. Zum Erreichen entsprechender Datenraten ist dann allerdings teure Hardware erforderlich, die mit ebenfalls mehreren Gigahertz takten kann.

Eine bessere Ausnutzung der Kabelbandbreite, und damit höhere Datenrate ergibt sich durch die Nutzung von höherwertigen Modulationsverfahren wie der 5-PAM z.B. beim Gigabit-Ethernet. Da für die Modulation jedoch zusätzliche Verarbeitungsschritte notwendig sind, steigt die Latenz der Datenübertragung an. Die zusätzliche Modulations-Hardware trägt nicht unerheblich zur Chip-Größe und zum Energiebedarf der Gesamtschaltung bei. Daher müssen Hardware-Aufwand, Datenrate und Energiebedarf gegeneinander abgewogen werden.

Schon aufgrund der Komplexität der Entwicklung analoger integrierter Schaltkreise, sind Modulationsverfahren attraktiv, die mit rein digitalen Schaltkreisen

implementierbar sind. Dazu zählen Modulationsformen, die sich auf die Zeitachse abbilden lassen, wie Pulsweitenmodulation und Pulsphasenmodulation. Vorteile der Pulsweitenmodulation sind eine niedrige Latenz und ein geringer Hardware-Aufwand. Dass die Pulsweitenmodulation eher ein Nischenprodukt für die Datenübertragung ist, liegt insbesondere an den niedrigen Datenraten bisheriger Implementierungen. Eine Ursache dafür liegt am Fokus der Implementierungen, und ist nicht grundsätzlich der Pulsweitenmodulation zuzuschreiben. Denn gerade auf Kanälen, auf denen die Bit-serielle Übertragung ihre potenten Endgeräte nicht ausnutzt, kann die Pulsweitenmodulation höhere Datenraten erreichen.

3 Aufbau, Funktionsweise und Programmierung von FPGAs

Field-Programmable Gate Arrays (FPGAs) sind Schaltkreise, die das Verhalten beliebiger anderer Schaltkreise nachbilden können. Das erfolgt einerseits mittels programmierbarer logischer Verknüpfungen (UND, ODER, ...) in konfigurierbaren Tabellen (englisch Look-Up Table, Abkürzung LUT), und andererseits durch eine beliebige Verbindung der Eingangs- und Ausgangssignale aller LUTs mittels flexibel kombinierbarer Leitungsstücke [KTR08]. Die Programmierung von FPGAs erfolgt mit herstellereigenen Programmen [Quartus13; ISE1212] Die Beschreibung der vom FPGA zu realisierenden Schaltung erfolgt mit Hardware-Beschreibungssprachen wie VHDL [Bha99] oder Verilog [Flü11]. Als flexible Hardware-Entwicklungsplattform sind FPGAs für die in dieser Arbeit vorgestellten Untersuchungen besonders geeignet.

Dieses Kapitel gibt einen kurzen Überblick über den Aufbau und die Funktionsweise von FPGAs. Dies beinhaltet eine Beschreibung der wichtigsten Schaltungsbestandteile und deren Kenngrößen. Ebenso wird der Arbeitsablauf bei der Entwicklung von FPGA-Schaltungen, vom VHDL-Quelltext bis zur Hardware dargestellt. Den Abschluss bilden einige Hinweise für die Erstellung asynchroner Schaltungsteile in FPGAs.

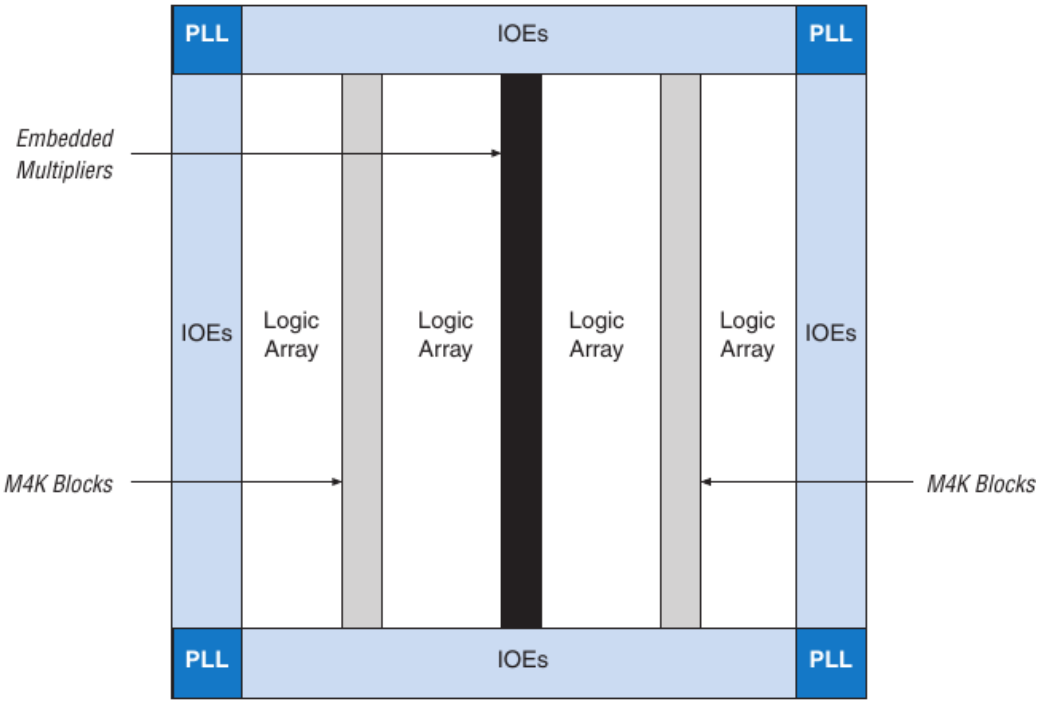

Abbildung 3.1: Der Aufbau eines Altera Cyclone II EP2C20 FPGA (aus dem Cyclone II Device Handbook [CycloneII08, S. 2-2]).

3.1 Aufbau eines FPGA am Beispiel des Cyclone II FPGA

Dieser Abschnitt beschreibt den Aufbau eines Cyclone II FPGA als typischen Vertreter von low-cost, SRAM-basierten FPGAs. Abbildung 3.1 zeigt die schematische Draufsicht auf die Siliziumfläche eines EP2C20 FPGA. Hauptbestandteile des FPGA sind Logikblöcke in den weißen Bereichen(Logic Array). Weitere Komponenten sind Ein-/Ausgabeblöcke (IOEs) am Rand, RAM-Blöcke (M4K) in den beiden grauen Spalten DSP-Blöcke (Embedded Multipliers) in der schwarzen Spalte und 4 PLLs in den Ecken. Die Verbindung der Funktionsblöcke erfolgt durch horizontal und vertikal verlaufende Leitungen. Die dazwischenliegenden Schaltkästen ermöglichen die Schaltung beliebiger Signalpfade innerhalb des FPGA. Die Verbindungsleitungen innerhalb des FPGA benötigen bis zu 80 % der Gesamtfläche.

Einen detaillierteren Blick in das FPGA bietet Abbildung 3.2. Darin ist ein Ausschnitt eines hypothetischen Cyclone II FPGA dargestellt. Erkennbar sind Spalten und Zeilen von Verbindungsleitungen. Die Schaltkästen an den Kreuzungspunkten der Leitungen dienen dem Programmieren der Leitungsverbindungen. Diese Leitungen und Schaltkästen werden auch als globale Routing-Ressourcen bezeichnet. Im FPGA sind weiterhin RAM-Blöcke und Logic-Array Blocks (LABs) spalten-

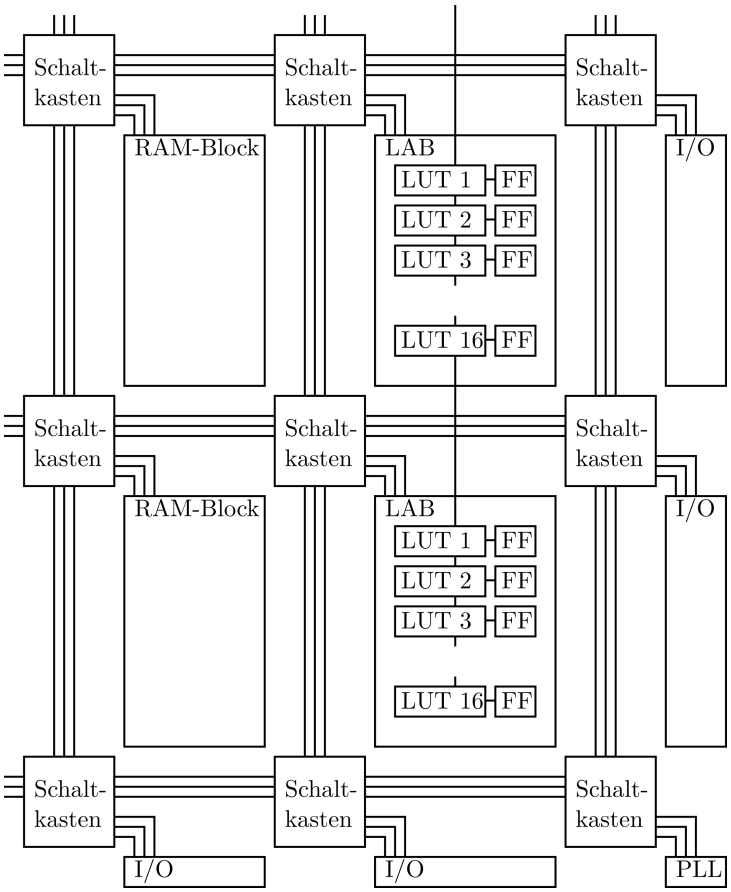

Abbildung 3.2: Ausschnitt aus der FPGA-Struktur.

weise angeordnet. Die Ein-/Ausgabelogik in Form von einzeln an die Schaltkästen angeschlossene Blöcke ist ebenfalls erkennbar. Die einzelnen Bestandteile des FPGA werden im Folgenden beschrieben.

3.1.1 Logic-Array Blocks (LABs)

Die Logikelemente (LE) sind, wie in Abbildung 3.2 zu sehen, in Blöcken zu 16 Look-up Tables (LUTs) und 16 D-Flipflops (FF) in den LABs zusammengefasst. Innerhalb der LABs können die Logikelemente beliebig „verdrahtet“ werden. Eine besondere Verbindung stellt dabei der Carry-Pfad dar. Er verbindet jede LUT mit der direkt unterhalb liegenden LUT, sogar über LAB-Grenzen hinweg. Der Name Carry-Pfad leitet sich vom ursprünglichen Zweck ab, eine möglichst schnelle Signalweiterleitung bei der Implementierung von Ripple-Carry-Addierern bereitzustellen.

a	b	c	d	combout
0	0	0	0	0
1	0	0	0	0
0	1	0	0	1
1	1	0	0	0
0	0	1	0	0
1	0	1	0	1
0	1	1	0	1
1	1	1	0	1
0	0	0	1	0
1	0	0	1	0
0	1	0	1	1
1	1	0	1	0
0	0	1	1	0
1	0	1	1	1
0	1	1	1	1
1	1	1	1	1

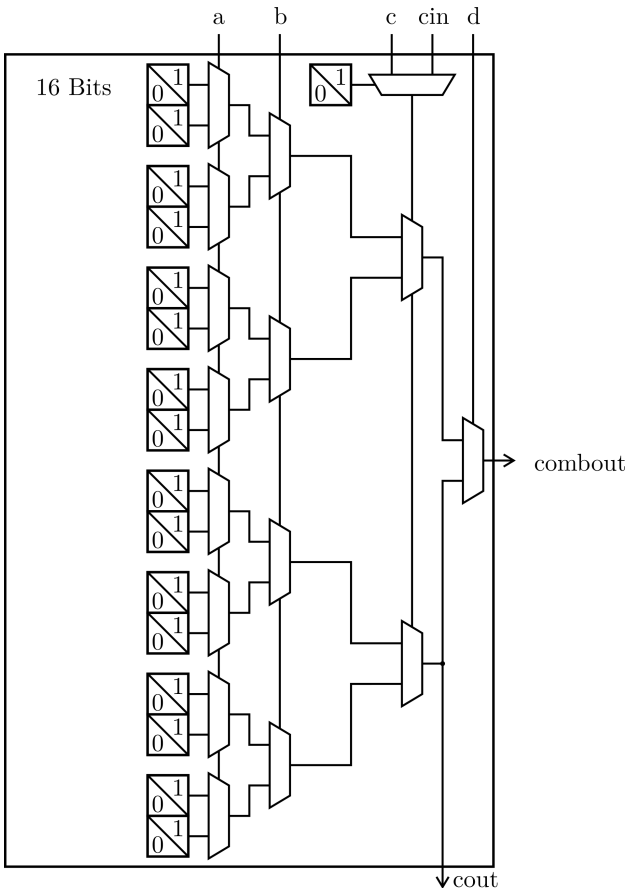

Abbildung 3.3: Wahrheitstabelle und Schaltbild einer typischen Look-up Table mit den Eingängen a bis d. Die 16 LUT-Bits entsprechen der Ergebnisspalte (`combout`) der Wahrheitstabelle. Zusätzlich ist ein wählbarer `cin`-Eingang, und ein `cout`-Ausgang vorhanden.

3.1.2 Look-up Tables (LUTs)

Die Realisierung der Logikfunktion innerhalb des FPGA erfolgt in den Look-up Tables (LUTs). Abbildung 3.3 zeigt eine Look-up Table (LUT) eines Cyclone II FPGA. Jede LUT realisiert eine Logikfunktion mit 4 Eingängen. Die 16 Ergebniswerte werden in SRAM-Bit gespeichert. Die vier logischen Eingangssignale (a bis d) adressieren durch Multiplexer einen der 16 Datenwerte des Speichers, und leiten diesen nach `combout`. Der Speicherinhalt wird z.B. durch das Aufstellen der Wahrheitstabelle bestimmt. Die dargestellte Look-up Table realisiert die Logikfunktion $combout = \neg a \wedge b \vee a \wedge c$.

Über die Grundfunktion der Wertetabelle hinaus bietet die FPGA-LUT weitere Funktionen. Diese Zusatzfunktionen werden durch weitere SRAM-Bits gesteuert.

Beispielhaft ist ein SRAM-gesteuerter Multiplexer dargestellt, der den Eingang der dritten Multiplexerspalte zwischen `c` und `cin` auswählt. Der Ausgang des unteren der beiden Multiplexer der dritten Spalte besitzt einen zweiten Abgriff namens `cout`. Dieser Ausgang ist direkt mit dem `cin`-Eingang der darunterliegenden LUT verbunden. Dieser relativ kurze Signalpfad durch die Look-up Table hat eine geringe Verzögerung zur Folge, und wird als Carry-Pfad bezeichnet. Im Cyclone II FPGA handelt es sich um die kleinste Signallaufzeit durch ein aktives Element. Neben der eigentlichen Intention, möglichst schnelle Ripple-Carry-Addierer zu implementieren, kommt diese Schaltungsweise dem Bau von asynchronen Zeitmessschaltungen, wie sie in Kapitel 4 vorgestellt werden, entgegen. Die vertikale Verbindung der Look-up Tables durch das Carry-Signal erstreckt sich über die gesamte FPGA-Spalte, also auch über die LAB-Grenzen und globalen Routing-Ressourcen hinweg. Die Anzahl der durch den Carry-Pfad verknüpfbaren Look-up Tables ist abhängig von der FPGA-Größe, und liegt im EP2C20 bei

$$16\ \text{LUT/LAB} \cdot 26\ \text{LAB} = 416\ \text{LUT},$$

da das FPGA 26 LABs pro Spalte aufweist.

3.1.3 Flipflops

Neben den Wertetabellen bieten FPGAs auch Flipflops (FF) als sequentielle Elemente. Die dedizierten Flipflops weisen eine geringere Fläche auf, und können mit einer höheren Taktfrequenz betrieben werden, als eine Implementierung mittels Look-up Tables. Die D-Flipflops des Cyclone II FPGA bieten eine Vielzahl von Einstellungen und Eingängen. Der interessierte Leser sei hier auf das Cyclone II Handbuch [CycloneII08, Seite 2-2 bis 2-6] verwiesen.

3.1.4 Block-RAM

Für das Speichern von Datenmengen im Kilo-Byte-Bereich stehen dedizierte RAM-Blöcke zur Verfügung. Im Cyclone II FPGA handelt es sich um 4 Kilo-Byte große Blöcke (M4K-Blöcke), die in ihren Daten- und Adressbreiten beliebig einstellbar sind. Bei größerem Speicherbedarf können mehrere M4K-Blöcke zu einem Speicher zusammengefasst werden.

3.1.5 Ein- und Ausgangselemente

Die Kommunikation mit der Außenwelt findet über die FPGA-Pins statt. Die Pins sind nicht direkt mit der FPGA-Schaltung verbunden. Die Ein- und Ausgangselemente (IOE) dienen der elektrischen Anpassung der FPGA-internen Signale an externe Logikpegel mit Spannungen bis zu 3,3 V. Die Ein-/Ausgabeelemente

des Cyclone II FPGA unterstützen mehrere Dutzend verschiedene I/O-Standards. Darunter sind Low-Voltage-TTL und CMOS mit Spannungen zwischen 1,8 V und 3,3 V, sowie LVDS und HSTL.

3.1.6 PLL

Im Cyclone II FPGA sind zwei oder vier Phasenregelschleifen, sogenannte Phase-Locked Loops (PLLs) enthalten. Sie ermöglichen aus einem Eingangstakt die gleichzeitige Erzeugung unterschiedlicher Takte für verschiedene synchrone Teilschaltungen auf dem FPGA. Im Rahmen dieser Arbeit dienen die PLLs der Einstellung einer beliebigen Systemfrequenz, ohne den extern angeschlossenen Oszillator wechseln zu müssen.

3.1.7 Weitere Spezialelemente

Innerhalb der verschiedenen FPGAs sind eine Reihe weiterer Blöcke implementiert. Im Cyclone II FPGA sind das beispielsweise Multiplizierer. Andere FPGAs bieten darüber hinaus Serialisierer-/Deserialisierer, oder ganze Mikroprozessoren. Diese Spezialelemente sind herstellerspezifisch, und werden in dieser Arbeit nicht verwendet.

3.1.8 Der Konfigurationsspeicher

Der Inhalt der SRAM-Zellen in den verschiedenen Blöcken des FPGA bestimmt deren Funktion. Die Konfiguration oder Programmierung des FPGAs besteht also aus dem korrekten Setzen aller SRAM-Bits im FPGA. Die Gesamtheit der SRAM-Bits wird als Bit-Stream bezeichnet. Der Bit-Stream bestimmt die Funktionalität des FPGA. Bei SRAM-basierten FPGAs wird der Bit-Stream beim Anschalten des FPGA von einem externen Konfigurationsspeicher geladen. Der folgende Abschnitt beschreibt die Wege zur Erzeugung eines Bit-Stream für ein Cyclone II FPGA.

3.2 Der Entwicklungsprozess von FPGA-Schaltungen

Dieser Abschnitt beschreibt die Werkzeuge zur Erstellung einer FPGA -Konfiguration für ein Cyclone II FPGA von Altera. Sämtliche Werkzeuge befinden sich in einem Software-Paket namens Quartus II. Im Rahmen dieser Arbeit kommen die Versionen Quartus II 12.1 patch 177 und Quartus II 13.0 sp1 für Linux, jeweils in der kostenpflichtigen subscription-edition (ca. 4000 US-$ jährlich) zum Einsatz.

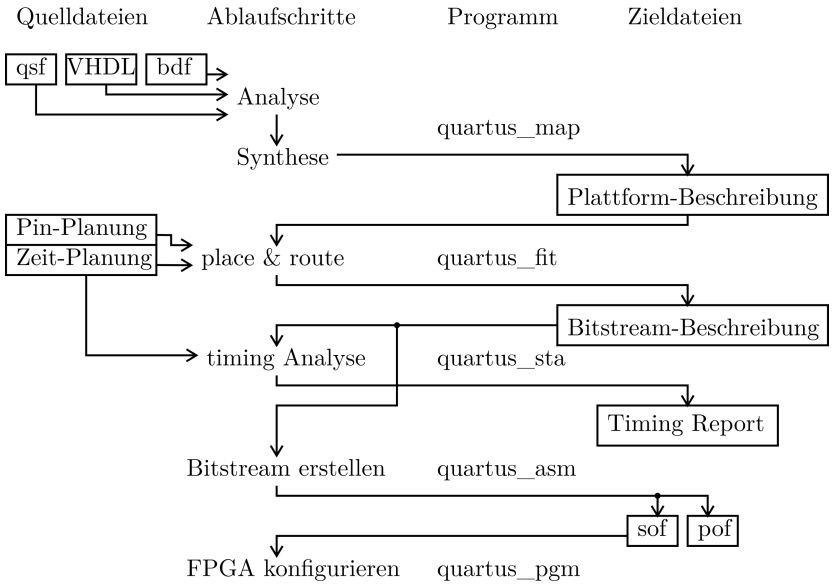

Abbildung 3.4: Programmablauf der Quartus-Synthese.

3.2.1 Der Quartus-Design-Flow

Die Schaltungsentwicklung mittels Quartus besteht aus sechs wesentlichen Schritten:

1. Erstellen der Hardware-Beschreibung.
2. Analyse der Hardware-Beschreibung und Abbilden der Funktion auf FPGA-spezifische Schaltungselemente.
3. Erzeugen der Bit-Stream-Beschreibung.
4. Analyse des Zeitverhaltens anhand der Bit-Stream-Beschreibung.
5. Erzeugen des Bit-Stream.
6. Konfigurieren des FPGA mit dem Bit-Stream.

Die letzten fünf Schritte werden durch jeweils ein Programm im Quartus-Programmpaket bearbeitet. Der Arbeitsablauf ist in Abbildung 3.4 dargestellt. Die Quelldateien stehen auf der linken Seite, die Ergebnisse des jeweiligen Arbeitsschrittes stehen auf der rechten Seite. Die Pfeile stellen die Abhängigkeiten der einzelnen Arbeitsschritte von den Eingabe- und Ausgabedaten dar. Im Folgenden werden die sechs Verarbeitungsschritte detailliert besprochen.

3.2.2 Die Hardware-Beschreibung

Am Beginn der FPGA-Entwicklung steht das Erstellen eines Projektes in Quartus.

```
quartus_sh --prepare -f CycloneII -d EP2C70F896C6 projekt
```

Dabei gibt `-f CycloneII` die FPGA-Familie an, und `-d EP2C70F896C6` legt das genaue FPGA fest. Quartus erstellt zwei Dateien `projekt.qpf` und `projekt.qsf`. Die qsf-Datei (Quartus Settings File) ist ein tcl-Skript, das sämtliche Projekteinstellungen enthält. In dieser Datei stehen Verweise auf alle verwendeten Quelldateien, wie VHDL-Quelltexte, verilog-Quelltexte, Schaltpläne (.bdf), Zeitvorgaben (.sdc), Prozessorspezifikation (.qsys), weitere Verweise auf Quelldateien (.qip) und unzählige mehr oder weniger dokumentierte weitere Formate. Für die Erzeugung dieser Quelldateien stehen eine Reihe von Werkzeugen zur Verfügung. Die erste Wahl ist in vielen Fällen die Grafische Oberfläche von Quartus II, die mit

```
quartus
```

gestartet wird. Aus Quartus heraus lassen sich die Programme Qsys und Megawizard starten, um vorgefertigte, parametrierbare Schaltkreisblöcke zu erstellen. Diese reichen von Schieberegistern über PLL-Konfigurationen bis hin zu ganzen Mikroprozessoren. Details sind der Dokumentation zur jeweiligen Quartus II Version zu entnehmen, z.B. [Quartus13]. Darüberhinaus bietet Quartus einen grafischen Schaltplaneditor und einen Texteditor für Quelltexte.

Nios II

Die Erstellung der Spezifikation (qsys-Dateien) von Mikroprozessoren erfolgt mit dem Programm

```
qsys-edit
```

Die Verwendung des Nios II-Prozessors bietet den Vorteil, dass Quartus einen Compiler und eine Entwicklungsumgebung mitbringt, die aus der Prozessorbeschreibung automatisch eine Hardware-Abstraktion mit passenden Treibern erstellt. Dadurch lassen sich Prozessorkomponenten, wie eine `jtag-uart` oder ein `epcs-controller` ohne Programmieraufwand hinzufügen. Im Anschluss kann zum Beispiel die `jtag-uart` als `stdin` und `stdout` im C-Quelltext für den Nios II verwendet werden.

3.2.3 quartus_map: Analyse & Synthese

Im diesem Schritt werden die vorhandenen Quelldateien analysiert und auf Korrektheit und Vollständigkeit überprüft. Das Kommando heißt

```
quartus_map --family=CycloneII projekt
```

Es erzeugt im aktuellen Verzeichnis unter anderem einen Ordner `db`, der alle für Quartus relevanten Informationen der Schaltung enthält. Alle weiteren Programme arbeiten mit Dateien in diesem Ordner.

3.2.4 quartus_fit: Place & Route

Das Programm

```
quartus_fit
```

ist der sogenannte Fitter, der die Platzierung und Verdrahtung vornimmt. Er erzeugt aus den vielen unterschiedlich beschriebenen Schaltungsteilen und den zugehörigen FPGA-spezifischen Zeit- und Pin-Beschreibungen eine zusammenhängende Netzliste (englisch netlist) für das Ziel-FPGA. Diese enthält die vollständige Konfiguration für das FPGA. Diese Beschreibung des Bit-Stream-Inhalts wird ebenfalls im Datenbankordner `db` gespeichert. Aufgrund der Vielzahl von Parametern und der prinzipiellen Komplexität des Problems nimmt dieser Schritt die meiste Zeit in Anspruch.

3.2.5 quartus_sta: Timing Analyse

Nachdem der Fitter lange Zeit versucht hat, die beschriebene Schaltung mit dem vorgegebenen Zeitverhalten umzusetzen, beantwortet der Static Timing Analyzer

```
quartus_sta
```

die Frage nach dem Erfolg. Die grundlegende Aussage ist die Taktfrequenz, mit der sich die erzeugte FPGA-Schaltung betreiben lässt. Ist diese Taktfrequenz niedriger als die vorgegebene Zieltaktfrequenz, so ist der Syntheselauf fehlgeschlagen. Abhängig von der Art der Schaltung und der Größe des Fehlers, müssen Eingabeparameter der vorherigen Schritte solange verändert werden, bis die Schaltung die gewünschte Geschwindigkeit erreicht.

3.2.6 quartus_asm: Bit-Stream erzeugen

Der Befehl

```
quartus_asm
```

erzeugt aus der im Datenbankordner `db` enthaltenen Information eine sof-Datei. Diese enthält unter anderem den Bit-Stream, so wie er an das FPGA gesendet wird.

3.2.7 quartus_pgm: Konfigurieren des FPGA

Das Programm

```
quartus_pgm
```

dient schlussendlich dazu den Bit-Stream in das FPGA zu laden.

3.3 Asynchroner Betrieb von FPGAs

An der Hardware-Struktur LUT→FF ist gut erkennbar, dass FPGAs für synchrone Schaltungen vorgesehen sind. Dafür bieten die Entwicklungswerkzeuge eine entsprechend gute Unterstützung. So erfordert die Analyse des Zeitverhaltens die Angabe einer Signalfrequenz für sämtliche Eingangssignale. Ebenso macht der Fitter seine Entscheidungen im Place & Route von den Zeitvorgaben abhängig. Problematisch wird diese Herangehensweise, sobald asynchrone Schaltungsteile in ihrem Zeitverhalten beschrieben werden müssen. Das Vokabular der Zeitbeschreibung ist auf synchrone Schaltungen gemünzt. Beispielsweise wird eine Taktangabe für jeden Eingangspin erwartet. Die Beschreibung asynchroner Schaltungen ist daher bestenfalls angenähert möglich.

Der Standardweg, um diese Probleme zu umgehen, ist die explizite Platzierung aller Bauteile asynchroner Schaltungen. Durch das Festlegen der Position der asynchron betriebenen Schaltungselemente werden implizit die Leitungsverzögerungen der kürzesten Verbindungsleitungen zwischen den Bauteilen vorgegeben. Damit ist auch das Zeitverhalten bestimmt.

Die explizite Platzierung hat eine Reihe von Nachteilen. Sie ist nur für ein FPGA-Modell gültig, und kann selbst auf diesem zu Konflikten mit anderen Logikressourcen führen. Für die explizite Platzierung ist es notwendig entsprechende Anweisungen in die qsf-Datei zu schreiben. Diese Platzierungsanweisungen beziehen sich auf Signalnamen, die üblicherweise erst nach einem Fitter-Durchlauf bekannt

sind, und sich auch mit den Optimierungseinstellungen ändern können. Daher sind manuelle Platzierungen eine inhärente Fehlerquelle bei Modifikationen an der FPGA-Schaltung.

Doch auch wenn die Platzierung eines asynchronen Schaltungsteils gelungen ist, wird der Fitter weiterhin versuchen, für die dort verwendeten Signale ein Zeitverhalten sicherzustellen, dass den funktionierenden Betrieb einer synchronen Schaltung ermöglicht. In Kapitel 6 werden mit den Anweisungen `set_false_path` und `set_disable_timing` Wege aufgezeigt, wie der Fitter entsprechende Signalpfade ignorieren kann.

Durch das Ignorieren bestimmter Signalpfade entsteht wiederum ein neues Problem: Da der Fitter keinen Überblick über die gesamte Schaltung mehr hat, kann er das Zeitverhalten nicht mehr sicher vorhersagen. Die Funktion der fertigen FPGA-Schaltung ist daher nicht immer gegeben, und muss in jedem Fall praktisch erprobt werden.

4 FPGA-basierte Zeitmesser

Die Nutzung von FPGAs als Erprobungsplattform für die PWM-Datenübertragung eröffnet eine Reihe von Möglichkeiten. Auf FPGAs ist die Implementierung verschiedener Zeitmesser möglich. Damit kann die Eignung unterschiedlicher Zeitmesser für die PWM-Datenübertragung praktisch erprobt werden. Dieses Kapitel erläutert zunächst die wichtigsten Kenngrößen elektronischer Zeitmesser und stellt zwei FPGA-basierte Zeitmesssysteme vor. Zuerst wird der synchrone Zähler vorgestellt. Als Vertreter der asynchronen Zeitmesser, wird im Anschluss die Tapped Delay-Line vorgestellt. Asynchrone Zeitmesser, wie die Tapped Delay-Line, bringen den Vorteil einer hohen Zeitauflösung. Sie erfordern jedoch aufgrund ihrer Implementierung eine Kalibrierung. Der zweite Teil des Kapitels geht auf die Erfordernisse und Möglichkeiten bei der Kalibrierung asynchroner Zeitmesser ein.

4.1 Der synchrone Zähler

Die Zeitmessung als solches ist eine verbreitete Messaufgabe, und in der Literatur [Alo+07; FC09; Wu09; Joo10; SK10; Wan+10; Wan+11; SHJ12; Szp+13; Mar+14] hinlänglich bekannt. Ein zu messendes Zeitintervall wird durch ein Startereignis und ein Stoppereignis bestimmt. Im gängigen Fall des synchronen, digitalen Zählers sind die beiden Ereignisse jeweils steigende Flanken auf dedizierten Start- und Stopplleitungen. Die zu messende Zeit wird mit einem bekannten Zeitintervall verglichen. Beim Zähler ist dieses Zeitintervall die Taktperiode $t_{\mathrm{clk}} = 1/f_{\mathrm{clk}}$, wobei f_{clk} die Taktfrequenz des Zählers ist.

4.1.1 Zeitgrößen

Messbereich

Der Messbereich des Zählers sagt aus, wie weit Startereignis und Stoppereignis auseinanderliegen dürfen. Bei einer Breite von N Bit kann der Zähler von 0 bis $2^N - 1$ zählen. Damit beträgt der Messbereich 0 s bis $(2^N - 1) \cdot t_{\mathrm{clk}}$. Der Messbereich des Zählers steigt also exponentiell mit der Breite des Zählers. Da die Zählerbreite bei einer FPGA-Implementierung beliebig wählbar ist, ist der Wertebereich

praktisch unbegrenzt. Beispielsweise benötigt die Darstellung des Alters des Universums in Picosekundenschritten

$$13{,}8 \cdot 10^9\,\mathrm{a}/10^{-12}\,\mathrm{s} \approx 4{,}355 \cdot 10^{17}\,\mathrm{s} \cdot 10^{12}\,\mathrm{s}^{-1} = SI4.355e29 \approx 0{,}69 \cdot 2^{99}$$

Schritte. Ein hypothetischer Zähler mit 100 Binärstellen, der mit einem Terahertz betrieben seit dem Urknall zählt, könnte noch weitere 26 Milliarden Jahre lang zählen. In Kombination mit dem Wissen, dass eine Zählertaktfrequenz von einem Terahertz mit den heutigen Atomgrößen nicht realisierbar ist, folgt, dass der Messbereich eines 100 Bit breiten Zählers für alle Zeitmessaufgaben ausreicht. Mit anderen Worten: Ein synchroner Zähler hat auf einem FPGA einen beliebig wählbaren Messbereich.

Auflösung

Die Auflösung eines Zählers beschreibt, wie fein die Schritte der Zeitmessung sind. Dabei wird die Auflösung direkt von der Taktfrequenz f_{clk} der Taktquelle bestimmt. Ein Takt von $f_{\mathrm{clk}} = 100\,\mathrm{MHz}$ hat eine Zeitauflösung von

$$t_{\mathrm{basis}} = 1/f_{\mathrm{clk}} = 10\,\mathrm{ns}$$

zur Folge. Der Takt des Zählers wird für das Erreichen kleiner Zeitschritte demzufolge so hoch wie möglich gewählt. Wie in jeder synchronen Schaltung darf die Taktperiode dabei nicht die Signallaufzeit des kritischen Pfades im Zähler unterschreiten. Die Länge des kritischen Pfades ist nahezu linear von der Breite des Zählers abhängig. Für eine gegebene Zeitauflösung existiert somit eine maximale, funktionsfähige Zählerbreite. Diese Breite ist Hardware-abhängig. Auf dem Cyclone II FPGA mit dem Speed Grade -6 erreicht ein 16 Bit-Zähler die Taktfrequenz $f_{\mathrm{clk},16} = 401{,}6\,\mathrm{MHz}$, und ein 64 Bit-Zähler erreicht $f_{\mathrm{clk},64} = 157{,}15\,\mathrm{MHz}$ (siehe [CycloneII08], Seite 5-15, Tabelle 5-15). Die Breite des Zählers muss demzufolge als Kompromiss aus dem Messbereich und der Zeitauflösung bestimmt werden. Die Zeitauflösung bestimmt ebenfalls den Quantisierungsfehler. Zählt der Zähler beispielsweise in 10 ns-Schritten, so verzählt er sich im Extremfall um knapp 10 ns. Daher beträgt der maximale Quantisierungsfehler ebenfalls 10 ns.

Präzision

Die Präzision eines Zeitmessers gibt an, wie gut die Reproduzierbarkeit einer Messung für die identische Signalfolge ist. Die Präzision eines Zählers ist von der Präzision der Zeitbasis, also der Taktquelle abhängig. Der Wert für die Schwankung der Taktperiode wird als Jitter bezeichnet. Für PLLs in FPGAs liegen die Angaben zu Jitter in der Größenordnung von 100 ps. Für das Cyclone II FPGA ist ein Maximalwert von 300 ps bei Taktfrequenzen über 100 MHz angegeben ([CycloneII08], Seite 5-66).

Unabhängig vom Jitter wird die Präzision auch durch den Quantisierungsfehler beschränkt. Für beliebige Zeiten sind Abweichungen bis zur Größe des Quantisierungsfehlers zu erwarten. Daher kann die Präzision eines 100 MHz-Zählers nie besser als 10 ns sein.

Genauigkeit

Als Genauigkeit wird die maximale Abweichung des Messwertes vom wahren Wert bezeichnet. Diese Beschreibung birgt jedoch zwei Probleme: Erstens muss der wahre Wert exakt bekannt sein, um die Abweichung eines Messwertes davon zu bestimmen. Und zweitens muss sichergestellt sein, dass die maximale Abweichung bestimmt wird. Beides ist prinzipiell unmöglich. In der Praxis wird die Genauigkeit daher abgeschätzt. Beim Zähler ist die Genauigkeit der Taktquelle entscheidend für die Genauigkeit des Zeitwertes. Das bedeutet, dass die Taktfrequenz f_{clk} mit der nominellen Frequenz $f_{\mathrm{osz.}}$ übereinstimmt, da die nominelle Frequenz für die Berechnung der Zeit genutzt wird. Eine Abweichung führt zu einem akkumulierenden Fehler, selbst wenn die Taktquelle sehr präzise mit der falschen Frequenz schwingt.

Pause zwischen zwei Messungen

Für die praktische Verwendung eines Zeitmessers ist ebenfalls bedeutsam, wie schnell nacheinander Messungen durchgeführt werden können. Die notwendige Messpause fällt abhängig vom Zeitmessverfahren unterschiedlich lang aus. Bei einem Zähler besteht dieses Problem nicht, da er innerhalb eines Taktes zurückgesetzt werden kann, damit die nächste Messung wieder bei Null beginnen kann.

4.1.2 Implementierungsrelevante Größen

Flächenbedarf

Der Flächenbedarf einer Zeitmessschaltung ist bei einer FPGA-Implementierung von Bedeutung. Einerseits muss die benötigte Anzahl Logikelemente verfügbar sein. Andererseits ist die Synthesedauer einer FPGA-Schaltung von der Anzahl der Logikelemente abhängig. Das wird bedeutsam, wenn viele verschiedene Schaltungsvarianten durchprobiert werden sollen. Beim Zähler wächst der Flächenbedarf der Bit-Register linear mit der Breite des Zählers, und logarithmisch mit dem Messbereich. Ein Zähler in einem FPGA benötigt je Bit-Stelle mindesten eine LUT für die logische Verknüpfung und ein Flipflop für das Speichern des Zähler-Bit.

Leistungsbedarf

Ein weiteres Kriterium für den praktischen Einsatz von Zeitmessschaltungen ist der Leistungsbedarf. Dieser setzt sich aus statischem $P_{\text{stat.}}$ und dynamischen Leistungsbedarf $P_{\text{dyn.}}$ zusammen. Der statische Leistungsbedarf des Zähler ist proportional zur Zählerbreite und damit zum Logarithmus des Messbereichs ($P_{\text{stat.}} \sim \ln t$). Der dynamische Leistungsbedarf des Zählers ist nahezu proportional zur Breite und zur Taktfrequenz des Zählers ($P_{\text{dyn.}} \sim \ln t \cdot f_{\text{clk}}$).

Fertigungstoleranzen

Die Fertigung elektronischer Bauteile unterliegt einer gewissen Streuung. Daraus folgt, dass nominell gleiche Bauteile leicht unterschiedliche Eigenschaften haben. Sehr deutlich tritt diese Bauteilstreuung bei der Leitfähigkeit von Halbleitern auf. Aus Sicht der digitalen Schaltungstechnik führen verschiedene Leitfähigkeiten zu unterschiedlichen Signallaufzeiten. Der synchrone Zähler ist von dieser Tatsache nicht betroffen, denn die Taktfrequenz wird stets so eingestellt, dass alle Signalpfade zwischen den Registern sicher innerhalb einer Taktperiode passiert werden. Synchrone Schaltungen sind daher gegen Fertigungstoleranzen „immun“, solange die Taktfrequenz einen Betrieb innerhalb der Toleranzen zulässt.

Temperatureinfluss

Bei der Zeitmessung mittels Zähler können Ungenauigkeiten dadurch entstehen, dass die Taktfrequenz des Quarzes nicht seiner nominellen Frequenz entspricht. Schwingquarze haben prinzipbedingt eine relativ große Temperaturabhängigkeit. Die logische Konsequenz ist, die Betriebstemperatur des Quarzes möglichst konstant zu halten. Lässt sich eine konstante Betriebstemperatur nicht sicherstellen, wird ein Herausrechnen des Fehlers durch Messen der Temperatur ermöglicht. Unter Verwendung der Temperaturkoeffizienten aus dem Datenblatt lässt sich für jede Temperatur die Schwingfrequenz des Quarzes bestimmen.

Betriebsspannung

Ein großer Vorteil des synchronen Zählers ist seine Toleranz gegenüber Betriebsspannungsschwankungen. Die Signallaufzeit innerhalb der Gatter ist signifikant von der Betriebsspannung abhängig, doch die Register „bremsen“ die Signale auf die Taktfrequenz.

Grenze der Zeitauflösung

Aufgrund der einfachen Implementierung und der günstigen Eigenschaften werden synchrone Zähler am weitaus häufigsten zur Zeitmessung eingesetzt. Wenn jedoch höhere Zeitauflösungen erforderlich sind, als ein Zähler erreichen kann, kommen in der Regel asynchrone Schaltungen zum Einsatz. Mit der Tapped Delay-Line, wird im Folgenden die Standardarchitektur für präzise Zeitmessung auf FPGAs vorgestellt.

4.2 Tapped Delay-Line

Die Tapped Delay-Line ist der Standardschaltkreis für FPGA-basierte präzise Zeitmesser [Szp+13]. Gründe dafür sind der einfache Aufbau mit digitalen Schaltkreisen sowie die einfache Auswertung der Messung.

Aufbau

Eine Tapped Delay-Line besteht aus einer Kette von Verzögerungselementen. In Abbildung 4.1 ist eine solche Verzögerungskette dargestellt. Ihre Elemente haben keine Signalverknüpfung zum Ziel, sondern verzögern das Signal um jeweils eine Gatterlaufzeit τ je Element. Nach jedem Verzögerungselement ist ein Abgriff zu einem Speicherelement vorhanden. Bei einer FPGA-Implementierung bieten sich als Speicherelemente D-Flipflops an. Das Startsignal wird in die Verzögerungskette eingespeist. Das Stoppsignal ist an den Takteingang aller D-Flipflops angeschlossen.

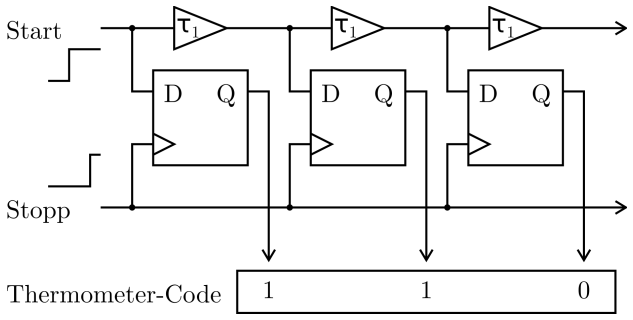

Abbildung 4.1: Tapped Delay-Line mit drei dargestellten Verzögerungsstufen.

Funktion

Die Funktion der Tapped Delay-Line beruht auf der Ausnutzung der Gatterlaufzeit integrierter Schaltkreise. Das Startsignal wird auf eine Kette von Signaltreibern geleitet. In endlicher Zeit passiert es einen Treiber nach dem anderen. Die Gatterlaufzeit dient als Zeitbasis für den Zeitmesser Tapped Delay-Line. Die Auswertung, wie weit das Startsignal bis zum Eintreffen des Stoppsignals gelangt ist, erfolgt durch D-Flipflops. Der Takteingang der D-Flipflops wird vom Stoppsignal gesteuert. Die Werte zwischen allen Verzögerungselementen werden durch die steigende Flanke des Stopp-Signals abgetastet. Der jeweilige Registerinhalt gibt dann an, ob die Flanke des Start-Signals das Verzögerungselement schon passiert hat. Das Ergebnis an den Registerausgängen ist eine Reihe von Einsen, gefolgt von einer Reihe Nullen. Der Ort des Eins-Null-Übergangs gibt an, welche Verzögerungselemente das Start-Signal bis zum Eintreffen des Stopp-Signals bereits passiert hat. Aus der Summe der Verzögerungszeiten τ_i der passierten Verzögerungselemente ergibt sich die Zeitdifferenz zwischen Start- und Stopp-Signal.

Auflösung

Die Zeitauflösung der Tapped Delay-Line ist um so besser, je kleiner die Verzögerungszeiten τ_i sind. Daher ist man bestrebt möglichst kleine Verzögerungen zu verwenden. In FPGAs dienen die Elemente entlang des dedizierten Carry-Pfades zwischen den LUTs als kleinste Verzögerung (detaillierter in Abbildung 3.1.2). Die Verzögerungszeit τ_i liegt zwischen 20 ps und 100 ps [FC09]. Ein digitaler Zähler müsste, zum Vergleich, eine Taktfrequenz zwischen 10 GHz und 50 GHz erreichen, was praktisch unmöglich ist.

Messbereich

Der Messbereich einer Tapped Delay-Line ist proportional zur Anzahl der Verzögerungsstufen. Soll beispielsweise der Messbereich verdoppelt werden, so sind auch doppelt so viele Verzögerungselemente erforderlich.

Flächenbedarf

Ebenso geht die Zeitauflösung linear in den Flächenbedarf ein.

$$A \sim T \cdot \frac{1}{\tau}$$

Bei einer Zeitauflösung von 100 ps sind für einen Messbereich von einer Sekunde 10^{10} Verzögerungselemente erforderlich. So große FPGAs wird es in den nächsten

Jahren nicht geben. Handhabbare Verzögerungsketten mit einigen hundert bis einigen tausend Verzögerungselementen erreichen Messbereiche im Bereich von 10 ns bis 100 ns. Beim Einsatz als Zeitmesser werden Tapped Delay-Lines häufig mit Zählern als groben Zeitmessern kombiniert, um den Messbereich zu vergrößern.

Leistungsbedarf

Der statische Leistungsbedarf ist proportional zur Fläche, und damit wie diese sowohl linear vom Messbereich, als auch von der Zeitauflösung abhängig.

$$P_{\text{stat.}} \sim t \cdot \frac{1}{\tau}$$

Die dynamische Leistungsaufnahme ist proportional zur Zeitauflösung, da bei höherer Auflösung mehr aktive Elemente in der gleichen Zeit umgeladen werden.

$$P_{\text{dyn.}} \sim \frac{1}{\tau}$$

Ein größerer Messbereich wirkt sich linear auf die dynamische Leistungsaufnahme aus, da mehr Elemente je Messung umgeladen werden. Ein grundlegender Unterschied zum Zähler ergibt sich jedoch aus der asynchronen Natur der Tapped Delay-Line. Die dynamische Leistung wird nur bei Messungen benötigt. Sie ist daher proportional zur Messrate. Im Gegensatz zum Zähler ist bei der Tapped Delay-Line die Zeitauflösung vom dynamischen Leistungsbedarf entkoppelt.

Fertigungstoleranzen und Präzision

Die vergleichsweise kleinen Zeitintervalle der Verzögerungskette offenbaren ein Problem: Da die Gatterverzögerungen herstellungsbedingten Streuungen unterworfen sind, sind auch die daraus resultierenden Zeiten τ_i ungleichmäßig. Die prozessbedingten Streuungen gehen dabei bis in den zweistelligen Picosekundenbereich, und sind damit in der gleichen Größenordnung, wie der zur Messung benutzte Verzögerungswert. Im Gegensatz zu einem Zähler, der ähnlich große Messfehler durch Jitter hat, ist die Auflösung der Tapped Delay-Line so groß, dass dieser Fehler signifikant wird. Tapped Delay-Lines erfordern daher immer eine (präzise) Kalibrierung, die den Verzögerungselementen Zeitwerte zuordnet, und damit die Fertigungsabweichungen quantifiziert.

Temperatur & Betriebsspannung

Neben den fertigungsbedingten Toleranzen, machen die Laufzeitschwankungen mit Temperaturveränderungen einen signifikanten Anteil des Messwertes aus. CMOS-Transistoren haben eine temperaturabhängige Leitfähigkeit. Daraus folgt, dass

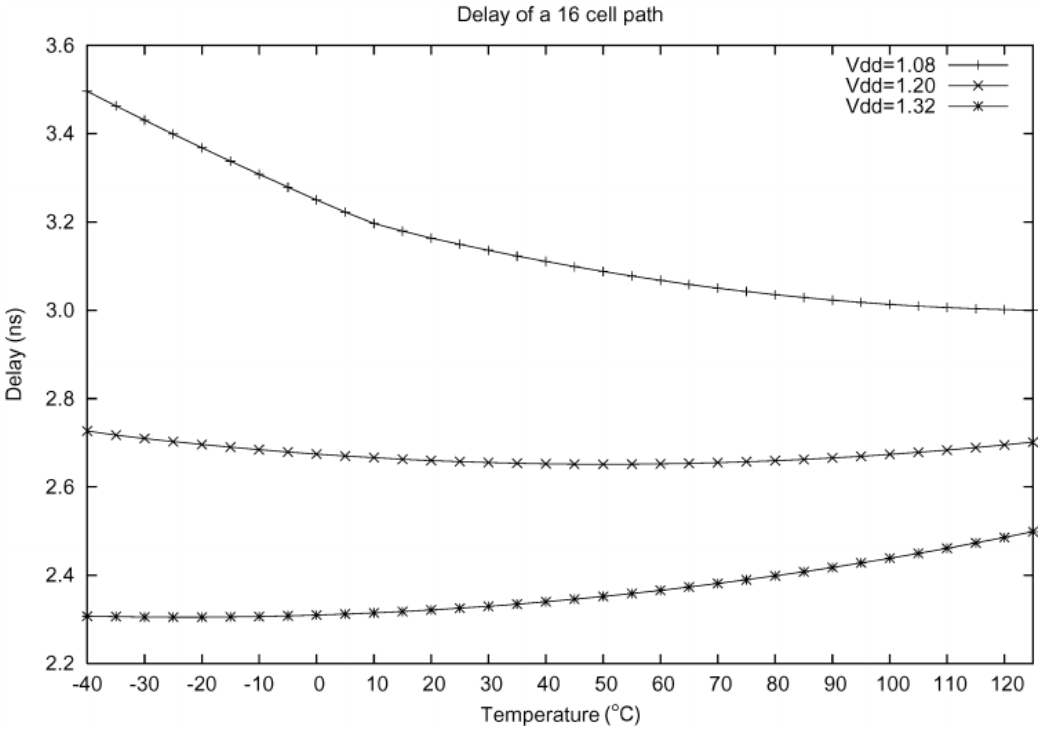

Abbildung 4.2: Aus [DH06]; die Abhängigkeit der Verzögerungszeit (Delay) eines Pfades mit 16 Gattern von Temperatur und Betriebsspannung (Vdd).

die Signallaufzeit der aktiven Buffer im FPGA abhängig von der Temperatur schwankt. Szplet, Kalisz und Jachna [SKJ09, Seite 10, Abbildung 14] haben gezeigt, dass Temperaturänderungen von 5 K Verzögerungsschwankungen $\Delta\tau_i$ im zweistelligen Picosekundenbereich zur Folge haben. Simulationen von Dasdan und Hom [DH06] zeigen, dass die Betriebsspannung ebenfalls einen Einfluss auf die Signallaufzeiten der Verzögerungselemente hat. Insbesondere kann sie Einfluss auf das Vorzeichen des Temperaturkoeffizienten haben, wie in Abbildung 4.2 dargestellt ist. Das genaue Zeitverhalten einer Schaltung ergibt sich demzufolge aus einer Kombination von Temperatureinfluss und temperaturabhängigem Spannungseinfluss.

4.3 Kalibrieren von FPGA-basierten Zeitmessern

Eine Kalibrierung asynchroner Zeitmesser ist notwendig, da die Verzögerungen der Verzögerungselemente unbekannt sind. Einerseits haben alle Verzögerungselemente aufgrund der Fertigungstoleranzen unterschiedliche Verzögerungswerte. Und andererseits hängen diese Verzögerungswerte von Versorgungsspannung, Temperatur

und Alter des Schaltkreises ab. Demzufolge muss eine asynchrone Zeitmessschaltung immer kalibriert werden. Je nach Schwankung der Umgebungsbedingungen muss diese Kalibrierung mehr oder weniger häufig erfolgen. Die Kalibrierung muss mit einer Methode erfolgen, die die erwünschte Genauigkeit erreicht. Die grundsätzliche Vorgehensweise ist folgende: Die Zeitmessschaltung misst bekannte Zeitintervalle. Die Ausgangswerte werden dann mit den bekannten Eingangswerten abgeglichen. Die verschiedenen Kalibrierverfahren unterscheiden sich in der Tatsache, wodurch die bekannten Zeitintervalle bekannt sind. Im Rahmen der Vorarbeiten für die Dissertation wurden neue Ansätze zur Kalibrierung von hochpräzisen Zeitmessern entwickelt [JHS13; JH15; HS16c; HS16a]. Aufgrund des Umfanges und weil die gewonnenen Erkenntnisse nur indirekt zu dieser Arbeit beitragen, werden an dieser Stelle nur einige grundlegende Eigenschaften dargestellt. Zwei einfache Kalibriermöglichkeiten sind die Laufzeitmessung und die statistisch gleichverteilte Messung, wie sie im Folgenden beschrieben werden.

4.3.1 Ausnutzung der Laufzeitdifferenzen verschiedener Leitungslängen

Eine grundsätzliche Schwierigkeit bei der Kalibrierung von Zeitmessern ist die Notwendigkeit Zeitdifferenzen in ausreichender Genauigkeit zu erzeugen. Dies ist beispielsweise durch die Signallaufzeit durch ein Leitungsstück bekannter Länge bei bekannter Ausbreitungsgeschwindigkeit gegeben. Eine Möglichkeit zur Erzeugung verschiedener Zeitdifferenzen ergibt sich durch die Nutzung einer Leitung mit einstellbarer Länge, wie z.B. eines Line-Stretcher [Microlab13]. Ein solcher Line-Stretcher ist ein Koaxialrohr, dass in seiner Länge von 696 mm bis 1001 mm variiert werden kann. Die Verzögerung des Line-Stretchers kann dadurch um

$$(1001\,\mathrm{mm} - 696\,\mathrm{mm})/c = 3{,}05 \cdot 10^{-1}\,\mathrm{m}/3 \cdot 10^{8}\,\mathrm{m\,s^{-1}} \approx 1 \cdot 10^{-9}\,\mathrm{s} = 1\,\mathrm{ns}$$

variiert werden. Die Bestimmung der Zeit kann somit durch eine Längenmessung erfolgen. Mit $1/c \approx 33\,\mathrm{ps\,cm^{-1}}$ sind im Millimeterbereich Auflösungen von einstelligen Picosekunden möglich. Wie Joost [Joo10] gezeigt hat, ist jedoch die Reproduzierbarkeit der Verzögerungszeiten schlecht. Durch die Mechanik des Line-Stretchers wirkt sich die mechanische Spannung der Arretierung auf das Zeitverhalten aus. Ebenso entsteht bei der Bewegung des Line-Stretchers eine mechanische Hysterese im Innern, die sich im zweistelligen Picosekundenbereich auswirkt. Die Kalibrierung mittels Line-Stretcher weist darüber hinaus weitere Nachteile auf: Der Line-Stretcher muss manuell eingestellt werden, und die Länge muss manuell dem jeweiligen Messwert zugeordnet werden. Dadurch ist eine regelmäßige automatische Kalibrierung bei Temperaturänderungen unmöglich. Desweiteren liegt die Dauer der Kalibrierung im Bereich einiger Minuten. Innerhalb dieser Zeit kann keine Messung stattfinden. Der nachfolgend beschriebene statistische Ansatz umgeht diese Nachteile.

4.3.2 Statistische Kalibrierung

Unter dem Namen „code density test“ haben Wu und Shi [WS08] einen statistischen Kalibrieransatz vorgestellt. Die Kalibrierung beruht auf der Tatsache, dass zwei unkorrelierte Oszillatoren mit verschiedenen Frequenzen beliebige Zeitdifferenzen erzeugen können. Wenn nun ein Oszillator das Start-Signal erzeugt, und der andere das Stopp-Signal, so sind innerhalb einer Taktperiode alle Zeitintervalle gleich wahrscheinlich. Bei einer Stoppsignalfrequenz von 400 MHz werden alle Zeitmessergebnisse von $2^{14} = 16384$ aufeinanderfolgenden Startsignalen mit einer Frequenz von 53,11 MHz addiert. Abbildung 4.3 zeigt die Zuordnung des Umschlagpunktes $0 \rightarrow 1$ anhand der Häufigkeit. Der Messbereich der Schaltung beträgt 2,5 ns. Dieser Messbereich nutzt die Elemente 4 bis 44 der Tapped Delay-Line. Die Häufigkeit des Auftretens der Ergebnisse lässt dabei einen Rückschluss auf die Breite der jeweiligen Zeitintervalle, also die Verzögerung der einzelnen Tapped Delay-Line-Elemente zu. In der Abbildung ist die unterschiedliche Häufigkeit durch die unterschiedlich hohen Stufen erkennbar. Die Auflösung der Kalibriermethode ist durch die Anzahl der Messungen beliebig wählbar, und liegt in diesem Fall bei $2{,}5\,\text{ns}/2^{14} \approx 0{,}156\,\text{ps}$.

Die Schaltung für die Kalibrierung kann vollständig in das FPGA integriert werden. Dadurch kann die Kalibrierung vollständig automatisiert werden, und es ist kein manueller Eingriff mehr notwendig. Entsprechend schnell erfolgt die Kalibrierung durch die statistische Auswertung innerhalb weniger Sekunden.

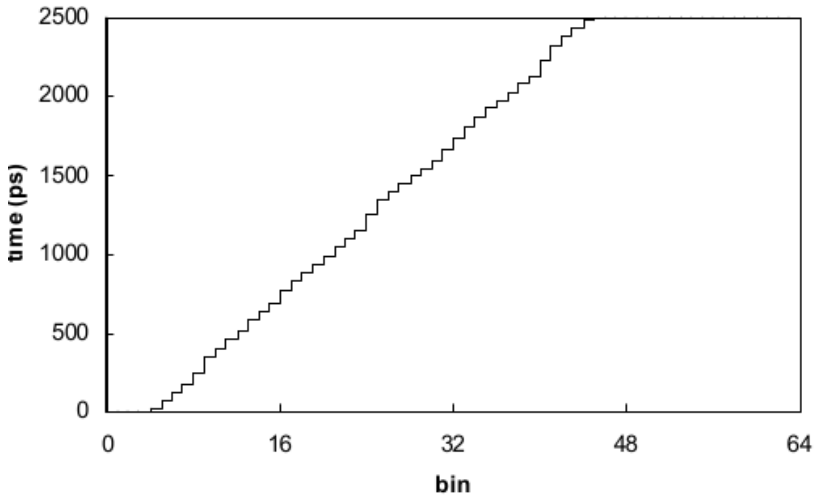

Abbildung 4.3: Die Verteilung des Tapped Delay-Line-Ausgangswertes (Anzahl der Einsen) auf den Messbereich von 2500 ps. Dargestellt ist die Position des $0 \rightarrow 1$-Übergangs.

5 Bandbreitenadaptive Datenübertragung mit Hilfe pulsbreitenmodulierter Signale

Dieses Kapitel umreißt die Kernidee der vorliegenden Dissertation: die Verwendung der Pulsweitenmodulation als Grundlage für die digitale Datenübertragung. Die Forschungsfrage lässt sich folgendermaßen formulieren: Kann durch Pulsweitenmodulation mit rein digitalen Schaltungen die Datenrate gegenüber der Bitseriellen Übertragung gesteigert werden? Die kurze Antwort ist: Ja. Die ausführliche Antwort liefern die nachfolgenden Kapitel. Dabei sind die drei wesentlichen Beiträge der vorliegenden Dissertation

1. die Verwendung asynchroner Modulationshardware zur Erhöhung der Zeitauflösung der PWM,
2. die Adaptivität an die (unbekannte) Kanalbandbreite und
3. die Senkung der Taktfrequenz bis auf die Pulsrate.

Erste Ergebnisse sind bereits veröffentlicht [SJH15a; HS16b] und mündeten in ein Patent [SJH15b].

Durch die Verwendung asynchroner Modulatoren und Demodulatoren erreicht die PWM-Datenübertragung Datenraten im Bereich von einem Gigabit pro Sekunde. Ein weiterer Vorteil der entwickelten Schaltungen ist die Adaptivität der Übertragungspulse an die Kabelbandbreite. Da die Anpassung ohne vorherige Kenntnis der Kabeleigenschaften erfolgt, können beliebige elektrische Leiter zur Signalübertragung eingesetzt werden. Kapitel 10 zeigt auf, inwieweit diese Adaptivität in Software realisiert werden kann, und ab wann und in welchem Umfang Hardware-Änderungen erforderlich sind. Dadurch wird insbesondere die breite Anwendbarkeit der vorgestellten PWM-Datenübertragungsmethode unterstrichen.

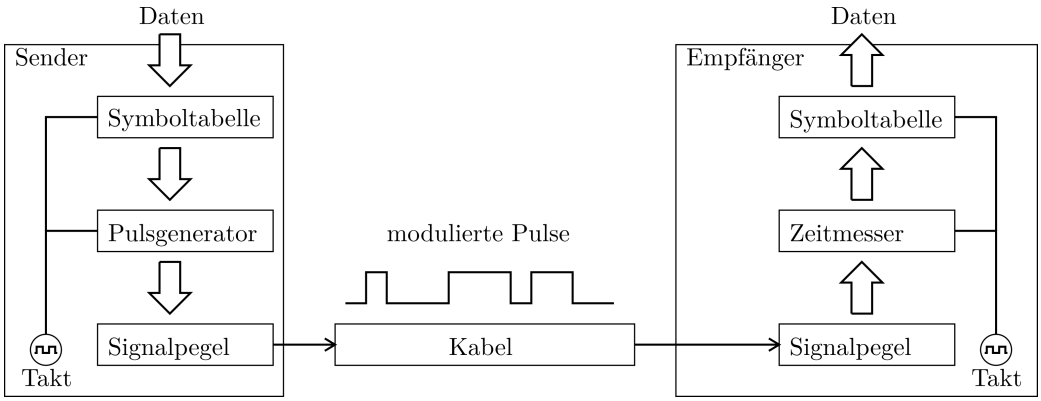

Abbildung 5.1: Grundsätzlicher Aufbau der PWM-Datenübertragungsstrecke.

5.1 Idee für die PWM-Datenübertragung

Der prinzipielle Aufbau der PWM-Datenübertragung, wie er in dieser Arbeit entstanden ist, ist in Abbildung 5.1 dargestellt. Die zu übertragenden Daten werden vom Sender über eine Symboltabelle in eine Pulslänge übersetzt. Die Pulslänge wird anschließend von einem Pulsgenerator in einen tatsächlichen Puls der vorgegebenen Dauer umgesetzt. Die Taktfrequenz des Senders bestimmt die Pulsrate. Daraufhin wird der gewünschte Signalpegel am Ausgangskontakt des Senders erzeugt. Dann legt der modulierte Puls auf dem Kabel die Strecke zum Empfänger zurück. Am Empfänger wird der ankommende Signalpegel als Puls erkannt. Der Zeitmesser bestimmt die Zeit zwischen steigender und fallender Flanke, also die Dauer des hohen Signalpegels. Aus dieser Zeit kann mit der Symboltabelle des Empfängers der Datenwert rekonstruiert werden.

Eigenschaften

Ein erfreuliches „Nebenprodukt“ der Verwendung der PWM ist, dass auf analoge Bauteile verzichtet werden kann. Alle Komponenten lassen sich in digitaler Hardware implementieren. Daraus ergibt sich eine Reihe von Vorteilen. Die Entwicklung analoger Schaltungsteile erfordert ein Vielfaches der Entwicklungszeit und Entwicklungskosten von digitalen Schaltungen. Weiterhin sinkt der Fertigungsaufwand des Schaltkreises, da bei analogen Schaltungen enge Toleranzgrenzen für einen bestimmungsgemäßen Betrieb notwendig sind. Außerdem müssen die analogen Schaltungsteile gegen Einflüsse aus digitalen Schaltkreisen geschützt werden. Auch sind Analogschaltungen auf CMOS-Schaltkreisen vergleichsweise groß, und benötigen entsprechend viel Energie. Zusammenfassend kann eine digitale Schaltung schneller und kleiner mit geringerem Energiebedarf gefertigt werden. Bei

der Verwendung von FPGAs kann sogar vollständig auf eine Schaltkreisfertigung verzichtet werden.

Die vorliegende Arbeit zeigt, dass die Pulsweitenmodulation eine energieeffiziente Methode zur Datenübertragung ist. Auf bandbreitenbegrenzten Verbindungen übertrifft die PWM-Datenübertragung die Datenrate Bit-serieller Übertragungen, weist dabei jedoch eine vergleichbare Latenz auf. Gleichzeitig lässt sich die Taktfrequenz von Sender und Empfänger bis auf die Pulsrate reduzieren. Eine Kalibrierung kann die Modulationsparameter adaptiv an Kanal, Sender und Empfänger anpassen. Aufgrund des geringen Hardware-Aufwandes eignet sich die PWM-Datenübertragung auch für Projekte mit stark limitierten Ressourcen.

Durch eine Implementierung in den Look-up Tables, kann das Übertragungssystem herstellerunabhängig in FPGAs implementiert werden. Damit kann die PWM-Datenübertragung zwischen FPGAs verschiedener Hersteller genutzt werden. Die offene Beschreibung, im Gegensatz zu fertigen Siliziumblöcken[1], lässt ein einfaches Nachimplementieren in ASICs, z.B. als Prozessorbrücke, zu.

Anwendungsbereiche

Ein möglicher Anwendungsbereich der PWM-Datenübertragung ist die Inter-Chip-Kommunikation über Platinengrenzen hinweg. Ebenso ist eine Verwendung im Bereich der Automation bis zu Entfernungen von einigen hundert Metern denkbar. Sinnvoll ist die Pulsweitenmodulation überall dort, wo die Bit-serielle Übertragung durch die gegebene Bandbreite begrenzt ist, aber aufwendigere Modulationsverfahren aufgrund zu hoher Latenzen oder eines zu hohen Energiebedarfs ungeeignet sind.

5.2 Teilprobleme

Beim Aufbau der PWM-Datenübertragungsstrecke ergeben sich drei Teilaufgaben:

1. Aufbau der asynchronen Pulserzeugung und Pulsmessung,
2. Implementierung der Symboltabellen und
3. Kalibrierung der Übertragungsstrecke.

Unter dem Gesichtspunkt der Realisierbarkeit einer solchen Schaltung sind darüberhinaus die Mechanismen der FPGA-Synthesewerkzeuge relevant. Weiterentwicklungen der Implementierung hinsichtlich der maximalen Datenrate, und der

[1] von den Herstellern als IP-Cores bezeichnet (Abk.: Intellectual Property)

Anpassungsfähigkeit an unbekannte Leitungen sind Teil der folgenden Untersuchungen. Nun folgt eine Übersicht der wesentlichen Ergebnisse innerhalb der sechs Problemfelder, mit Verweisen auf die dazugehörigen Kapitel dieser Arbeit.

5.2.1 Die Erhöhung der Zeitauflösung mit asynchroner Logik

Die Zeitauflösung der Endgeräte ist ausschlaggebend für die erreichbare Datenrate. Synchrone Schaltungen erreichen Taktraten im einstelligen Gigahertzbereich. Bei 4 GHz wird eine Zeitauflösung von 250 ps erreicht. Signifikant höhere Zeitauflösungen sind nur durch die Abkehr vom synchronen Schaltungsbetrieb möglich. Im Bereich der präzisen Zeitmesser (siehe Kapitel 4) sind asynchrone Schaltungen der Stand der Technik. Weiterhin haben Costinett, Rodriguez und Maksimovic [CRM13] einen FPGA-basierten, digitalen Pulsbreitenmodulator vorgestellt, der Zeitauflösungen im Bereich von 100 ps erreicht. In dieser Arbeit wird der Modulator dahingehend modifiziert, dass keine FPGA-Modell-spezifischen expliziten Platzierungen mehr notwendig sind. Die asynchrone Pulserzeugung im Sender, wird um eine asynchrone Zeitmessung im Empfänger ergänzt, und daraus eine PWM-Datenübertragungsstrecke entwickelt. Kapitel 6 vergleicht verschiedene Implementierungsansätze für Pulsgenerator und Zeitmesser im Hinblick auf die Anwendung zur Datenübertragung, und entwickelt an die Datenübertragung angepasste Lösungen. Kapitel 7 beschreibt einen Machbarkeitsnachweis, mit der erstmalig eine asynchrone PWM-Datenübertragung zwischen zwei FPGAs gelungen ist. Die Ergebnisse in Kapitel 8 zeigen eine Datenrate um 70 MBit/s, und deuten damit trotz nicht optimaler Parameter das Potential der asynchronen Implementierung der PWM-Datenübertragung an.

5.2.2 Die Symboltabellen in Sender und Empfänger

Die Verwendung von Pulsweitenmodulation für die digitale Datenübertragung bietet eine Reihe wählbarer Parameter. Hierzu gehört unter anderem die Anzahl der Einträge in der Symboltabelle n_{Symbol}. Um hier Anhaltspunkte für die optimale Wahl der Parameter zu erhalten, erfolgt zunächst in Abschnitt 5.3 eine mathematische Betrachtung der PWM-Datenübertragung.

Die Implementierung der beiden Symboltabellen ist eine Standardaufgabe im FPGA-Entwurf. Es muss nur darauf geachtet werden, dass die Tabellen dynamisch angepasst werden können. Hierfür eignen sich insbesondere die FPGA-internen RAM-Blöcke.

5.2.3 Kalibrierung der asynchronen Übertragungsstrecke

Durch die Verwendung asynchroner Schaltungsteile ergibt sich die Notwendigkeit einer Kalibrierung der Datenübertragungsstrecke, um die Zuordnung von Datenwerten zu Pulslängen herzustellen. Für den Prototyp wird in Abschnitt 7.3 eine einfache Kalibriermethode vorgestellt, die keine externe Hardware benötigt.

5.2.4 FPGA-Synthesewerkzeuge denken synchron

Ein wiederkehrendes Problem bei der Umsetzung der in dieser Arbeit vorgestellten Schaltungen stellen die Optimierungsversuche des Synthesewerkzeugs Quartus dar. Aufgrund der Ausrichtung der FPGA-Entwicklung auf synchrone Schaltungen ist es außerordentlich schwierig, reproduzierbar asynchrone Schaltungsteile zu erzeugen. Bei der Erläuterung der asynchronen Schaltungsteile werden die jeweils notwendigen Vorkehrungen zum Schutz der Schaltung vor dem Synthesewerkzeug beschrieben. Abschnitt 11.2 macht Vorschläge zur Verbesserung der FPGA-Synthesewerkzeuge im Hinblick auf asynchrone Schaltungen.

5.2.5 Die maximal erreichbare Datenrate

Die Fragestellung, welche Datenrate mit der PWM-Datenübertragung möglich ist, wird auf zwei Wegen untersucht. Abschnitt 5.3 stellt die mathematischen Zusammenhänge dar, und leitet daraus die theoretisch optimale Datenrate ab. Kapitel 9 nähert sich diesem Problem von der praktischen Seite, und zeigt den Aufbau einer PWM-Datenübertragung mit einer Datenrate von 1 GBit/s. Bemerkenswert an der dafür verwendeten Schaltung ist, dass Sender und Empfänger jeweils nur mit 333 MHz betrieben werden.

5.2.6 Anpassung an lange, unbekannte Kabel

Kapitel 10, das letzte Implementierungskapitel dieser Arbeit, widmet sich der PWM-Datenübertragung über ein langes Kabel. Hierfür werden Twisted-Pair-Kabel bis zu einer Länge von 223,2 m untersucht. Dabei ist einerseits das Verhalten der in Kapitel 7 vorgestellten Schaltung von Belang, andererseits fließen neue Architekturideen in die Schaltung ein, die für geringe Bandbreiten eine bessere Ausnutzung der FPGA-Ressourcen ermöglicht.

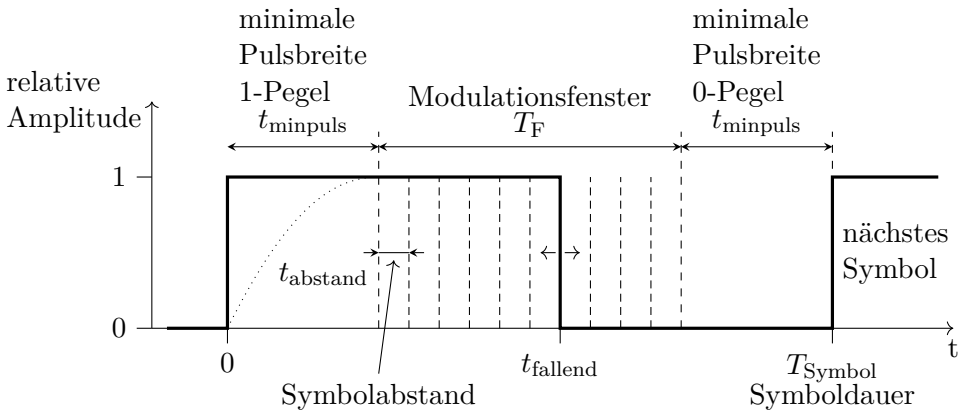

Abbildung 5.2: Ein Pulsweitenmodulation-Symbol im Detail.

5.3 Bezeichnung der Größen am Puls

Die Pulsweitenmodulation nutzt ein binäres Signal mit den normierten Pegeln 0 und 1. Ein Symbol besteht aus einem Puls vom Zeitpunkt 0 bis zum variablen Zeitpunkt t_{fallend}, gefolgt von einer Zeit, in der das Signal auf dem 0-Pegel bleibt. Wie in Abbildung 5.2 dargestellt, besteht jedes Symbol aus drei Teilen. Der erste Teil ist die steigende Flanke zum Zeitpunkt 0, gefolgt von der minimalen Dauer des 1-Pegels. Danach schließt sich, als zweiter Teil, das Modulationsfenster T_{F} an. Innerhalb dieses Zeitraums tritt die fallende Flanke des Pulses auf. Der dritte Teil ist die minimale Dauer des 0-Pegels. Der nächste Puls beginnt nach der Symboldauer T_{Symbol}. Die Pulsdauer oder auch Pulsbreite des 1-Pegels repräsentiert den Datenwert.

5.3.1 Die Bedeutung der minimalen Pulsdauer

Die idealisierte Darstellung als Rechtecksignal vernachlässigt die Bandbreitenbegrenzung durch den Kanal. Ein möglicher, realer Verlauf der steigenden Flanke ist in Abbildung 5.2 gepunktet dargestellt. Der Effekt der endlichen Bandbreite findet dadurch Berücksichtigung, dass eine minimale Pulsdauer t_{minpuls} nicht unterschritten werden darf. Nach der Zeit t_{minpuls} ist ein Signalwechsel vollzogen. Diese Zeit betrifft demzufolge beide Flanken. Sowohl High-Pegel als auch Low-Pegel dürfen die minimale Pulsbreite t_{minpuls} nicht unterschreiten. Im Vergleich mit einer Bit-seriellen Übertragung über den gleichen Kanal, entspricht die minimale Pulsdauer der kleinsten Bit-Dauer. Diese minimale Pulsdauer ist für eine Übertragungsstrecke fest.

5.3.2 Das Modulationsfenster

Innerhalb des Modulationsfensters beginnt die fallende Flanke. Da die Berücksichtigung der Kanalbandbreite schon durch die minimale Pulsdauer geschehen ist, kann innerhalb des Modulationsfensters die Zeitauflösung t_{abstand} der Hardware voll ausgenutzt werden. Je besser die Zeitauflösung ist, desto mehr Symbole können innerhalb des Modulationsfensters unterschieden werden. Aus der Anzahl der Symbole n_{Symbol} und dem Symbolabstand t_{abstand} ergibt sich die Länge des Modulationsfensters

$$T_{\text{F}} = t_{\text{abstand}} \cdot (n_{\text{Symbol}} - 1)\,.$$

Die Unterscheidung verschiedener Pulslängen

Der Empfänger unterscheidet verschiedene Symbole anhand ihrer Pulsdauer. Die Zeitauflösung von Sender und Empfänger bestimmt, wie dicht die Pulsdauern verschiedener Symbole beieinander liegen dürfen. Dieser Abstand t_{abstand} (in Abbildung 5.2 gestrichelt) kann sehr viel kleiner sein als die minimale Pulsdauer. Das ist eine *wesentliche* Eigenschaft der Pulsweitenmodulation. Der Parameter t_{abstand} geht linear in die Breite des Modulationsfensters ein.

Die Anzahl unterschiedlicher Symbole

Die Symbolanzahl n_{Symbol} ist bei der PWM-Datenübertragung ein frei wählbarer Parameter. Sie gibt an, wie viele verschiedene Zeitpunkte t_{fallend} für die fallende Flanke innerhalb des Modulationsfensters möglich sind. Die Wahl entscheidet über die Anzahl $b = \log_2 n_{\text{Symbol}}$ Bit die je Symbol übertragen werden. Aus dieser Sicht ist ein möglichst großes b wünschenswert. Andererseits geht die Symbolanzahl auch in die Länge des Modulationsfensters, und damit in die Symboldauer T_{Symbol} ein. Die Symboldauer ergibt sich, wie in Abbildung 5.2 ersichtlich, aus der Summe der minimalen Pulsdauer für High- und Low-Pegel und dem Modulationsfenster T_{F}:

$$T_{\text{Symbol}} = t_{\text{minpuls}} + t_{\text{minpuls}} + T_{\text{F}} = 2 \cdot t_{\text{minpuls}} + t_{\text{abstand}} \cdot (n_{\text{Symbol}} - 1)\,. \quad (5.1)$$

Eine kürzere Symboldauer ermöglicht die Übertragung von mehr Symbolen in der gleichen Zeiteinheit. Eine kurze Symboldauer T_{Symbol} steht jedoch einer großen Anzahl verschiedener Symbole n_{Symbol} entgegen. Im Folgenden wird sowohl eine praktische Lösung, als auch das mathematisches Optimum hergeleitet.

5.4 Berechnung der PWM-Datenrate

Dieser Abschnitt bietet zuerst eine Abschätzung der optimalen Symbolanzahl und der daraus resultierenden Datenrate anhand beispielhafter Werte. Im Anschluss erfolgt die mathematische Herleitung der optimalen Datenrate.

Die Anzahl unterscheidbarer Symbole beeinflusst sowohl die Anzahl Datenbit b je Symbol, als auch die Symbollänge T_{Symbol}. Die Symbolanzahl bestimmt über den Zusammenhang

$$b = \log_2 n_{\text{Symbol}} \iff 2^b = n_{\text{Symbol}} \tag{5.2}$$

die Anzahl übertragener Datenbit b pro Symbol. Die Datenrate R ist eine Funktion von der Symbolanzahl n_{Symbol}, und ergibt sich als Quotient

$$R = \frac{b}{T_{\text{Symbol}}} \tag{5.3}$$

Durch Einsetzen der Gleichungen (5.1) und (5.2) in Gleichung (5.3) ergibt sich für die Datenrate

$$R(b) = \frac{b}{2 \cdot t_{\text{minpuls}} + t_{\text{abstand}} \cdot (2^b - 1)} = \frac{\log_2 n_{\text{Symbol}}}{2t_{\text{minpuls}} + t_{\text{abstand}} \cdot (n_{\text{Symbol}} - 1)}. \tag{5.4}$$

Der Verlauf der Datenrate in Abhängigkeit der Bit-Anzahl ist in Abbildung 5.3 dargestellt. Die Darstellung ist ein Beispiel für einen 25 MHz-Kanal mit Werten von $t_{\text{minpuls}} = 20\,\text{ns}$ für die minimale Pulsbreite und $t_{\text{abstand}} = 2\,\text{ns}$ für den Symbolabstand. Aus der Abbildung 5.3 ist ersichtlich, dass die maximale Datenrate knapp unter 60 MBit/s beträgt. Die dafür notwendigen optimale Symbolanzahl liegt zwischen 8 Symbolen (3 Bit/Symbol) und 16 Symbolen (4 Bit/Symbol). Bei $b = 4\,\text{Bit}$ ist die Datenrate mit

$$R = \frac{4\,\text{Bit}}{2 \cdot 20\,\text{ns} + 2\,\text{ns} \cdot (2^4 - 1)} = \frac{2}{35}\text{Bit/ns} \approx 57{,}1\,\text{MBit/s}$$

nahe am theoretischen Optimum von $R_{\text{max}} \approx 57{,}38\,\text{MBit/s}$.

5.4.1 Herleitung der optimalen Bit-Anzahl je Symbol

Für die Abschätzung der Leistungsfähigkeit eines Systems mit gegebenem Symbolabstand t_{abstand} und gegebener minimaler Pulsdauer t_{minpuls} mag diese Herangehensweise ausreichend sein. In der Praxis ist jedoch meist eine zu erreichende Datenrate vorgegeben. Dann erst wird ein Übertragungsverfahren ausgewählt, dass die vorgegebene Datenrate innerhalb bestimmter Parameter erreichen kann. Im Falle der Pulsweitenmodulation gehören die minimale Pulsdauer t_{minpuls} und die Zeitauflösung t_{abstand} zu diesen Parametern.

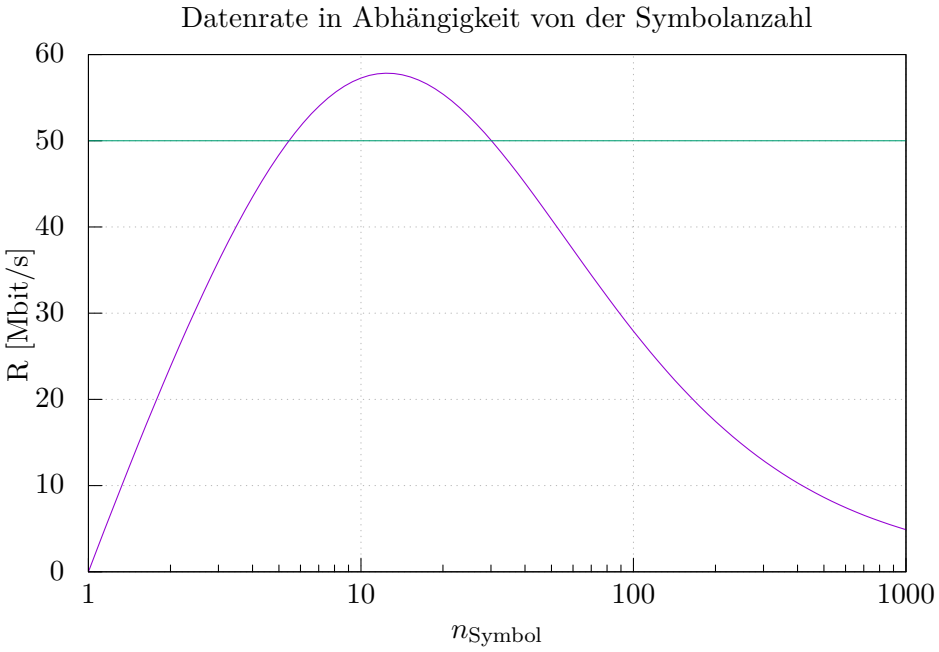

Abbildung 5.3: In violett, die Datenrate der PWM-Datenübertragung bei einer minimalen Pulsbreite $t_{\text{minpuls}} = 20\,\text{ns}$ und einer Signalauflösung $t_{\text{abstand}} = 2\,\text{ns}$. Das theoretische Maximum beträgt $R = 57{,}38\,\text{MBit/s}$ bei $n_{\text{Symbol}} = 12{,}47$ Symbolen. Die grüne Linie bei 50 MBit/s entspricht der maximalen Datenrate einer Bit-seriellen Übertragung mit idealem Tiefpass.

Zur übersichtlicheren Darstellung dienen folgende Identitäten: Die Symbolanzahl bestimmt über den Zusammenhang

$$x = n_{\text{Symbol}} = 2^b$$

die Anzahl übertragener Datenbit b pro Symbol.

Die Symboldauer wird folgendermaßen abgekürzt:

$$g(x) = T_{\text{Symbol}} = 2 \cdot t_{\text{minpuls}} + t_{\text{abstand}} \cdot (x - 1).$$

Eingesetzt in Gleichung (5.3) lautet die Formel für die Datenrate

$$R(x) = \frac{b}{T_{\text{Symbol}}} = \frac{\log_2 x}{g(x)}. \tag{5.5}$$

Die optimale Symbolanzahl x wird durch Ableiten und Nullsetzen $dR/dx = 0$ bestimmt. Eine übersichtlichere Darstellung lässt sich durch Ableiten des Reziproken

der Datenrate $1/R$ erreichen. Die so festgelegte gemittelte Bit-Dauer $t_b = 1/R$ besitzt, aufgrund der Stetigkeit für $x > 0$, die an den selben x-Werten Extremwerte wie R. Die Formel für die Bit-Dauer lautet

$$t_b = \frac{1}{R} = \frac{g(x)}{\log_2 x}. \tag{5.6}$$

Nach Einsetzen von $\log_2(x) = \frac{\ln x}{\ln 2}$ folgt

$$t_b = \frac{g(x)}{\frac{\ln x}{\ln 2}} = \frac{\ln 2}{\ln x} \cdot g(x).$$

Die Ableitung lautet

$$\frac{dt_b}{dx} = \frac{d}{dx}\left(\frac{\ln 2}{\ln x} \cdot g(x)\right).$$

Das Ausklammern von $\ln 2$ liefert

$$\frac{dt_b}{dx} = \ln 2 \cdot \frac{d}{dx}\left(\frac{1}{\ln x} \cdot g(x)\right).$$

Nach der Produktregel gilt

$$\frac{dt_b}{dx} = \ln 2 \cdot \left(\frac{d\left(\frac{1}{\ln x}\right)}{dx} \cdot g(x) + \frac{1}{\ln x} \cdot \frac{d\left(g(x)\right)}{dx}\right). \tag{5.7}$$

Die Ableitung von $\frac{1}{\ln x}$ ergibt

$$\frac{d\frac{1}{\ln x}}{dx} = \frac{d\frac{1}{\ln x}}{d\ln x} \cdot \frac{d\ln x}{dx} = \frac{d\frac{1}{u}}{du} \cdot \frac{d\ln x}{dx} = -\frac{1}{u^2} \cdot \frac{1}{x}. = -\frac{1}{xu^2}. = -\frac{1}{x\left(\ln x\right)^2}. \tag{5.8}$$

Eingesetzt folgt:

$$\frac{dt_b}{dx} = \ln 2 \cdot \left(-\frac{1}{x(\ln x)^2} \cdot g(x) + \frac{1}{\ln x} \cdot \frac{d\left(g(x)\right)}{dx}\right).$$

Durch Ausklammern von $\frac{1}{\ln x}$ ergibt sich

$$\frac{dt_b}{dx} = \frac{\ln 2}{\ln x} \cdot \left(-\frac{1}{x\left(\ln x\right)} \cdot g(x) + \frac{d\left(g(x)\right)}{dx}\right).$$

Die verbleibende Ableitung ist

$$\frac{d\left(g(x)\right)}{dx} = \frac{d}{dx}\left(2 \cdot t_{\text{minpuls}} + t_{\text{abstand}} \cdot (x-1)\right) = t_{\text{abstand}}.$$

Deren Lösung t_{abstand} eingesetzt ergibt

$$\frac{dt_b}{dx} = \frac{\ln 2}{\ln x} \cdot \left(-\frac{1}{x\,(\ln x)} \cdot g(x) + t_{\text{abstand}} \right).$$

Nach dem Einsetzen von $g(x)$ und dem Erweitern des letzten Summanden folgt

$$\frac{dt_b}{dx} = \frac{\ln 2}{\ln x} \cdot \left(-\frac{1}{x\,(\ln x)} \cdot (2 \cdot t_{\text{minpuls}} + t_{\text{abstand}} \cdot (x-1)) + \frac{t_{\text{abstand}} \cdot x \ln x}{x \ln x} \right).$$

Nun wird $\frac{1}{x \ln x}$ ausgeklammert

$$\frac{dt_b}{dx} = \frac{\ln 2}{x\,(\ln x)^2} \cdot (-\,(2 \cdot t_{\text{minpuls}} + t_{\text{abstand}} \cdot (x-1)) + t_{\text{abstand}} \cdot x \ln x),$$

innerhalb der Klammer ausmultipliziert

$$\frac{dt_b}{dx} = \frac{\ln 2}{x\,(\ln x)^2} \cdot (-2t_{\text{minpuls}} - t_{\text{abstand}} x + t_{\text{abstand}} + t_{\text{abstand}} \cdot x \ln x),$$

umgestellt und wieder zusammengefasst.

$$\frac{dt_b}{dx} = \frac{\ln 2}{x\,(\ln x)^2} \cdot (t_{\text{abstand}} \cdot (x \cdot (\ln x - 1) + 1) - 2t_{\text{minpuls}}). \tag{5.9}$$

Der Extremwert wird durch Nullsetzen bestimmt.

$$0 = \frac{\ln 2}{x\,(\ln x)^2} \cdot (t_{\text{abstand}} \cdot (x \cdot (\ln x - 1) + 1) - 2t_{\text{minpuls}}).$$

Unter der (sinnvollen) Annahme, dass $x > 1$ gilt, genügt eine Betrachtung des zweiten Faktors.

$$0 = t_{\text{abstand}} \cdot (x \cdot (\ln x - 1) + 1) - 2t_{\text{minpuls}}. \tag{5.10}$$

Die so erhaltene Gleichung (5.10) lässt sich nicht nach x umstellen. Durch Trennen der Parameter t_{abstand} und t_{minpuls} von x wird die Gleichung aber noch etwas übersichtlicher. Zunächst wird $2t_{\text{minpuls}}$ addiert

$$2t_{\text{minpuls}} = t_{\text{abstand}} \cdot (x \cdot (\ln x - 1) + 1).$$

Die anschließende Division durch $2t_{\text{abstand}}$ liefert mit

$$v = \frac{t_{\text{minpuls}}}{t_{\text{abstand}}} = \frac{1}{2} \cdot (x \cdot (\ln x - 1) + 1)$$

das Ergebnis, in das nun noch x eingesetzt wird:

$$v = \frac{t_{\text{minpuls}}}{t_{\text{abstand}}} = \frac{1}{2} \cdot (n_{\text{Symbol}} \cdot (\ln n_{\text{Symbol}} - 1) + 1). \tag{5.11}$$

In dieser Form lassen sich aus der Gleichung einige allgemeingültige Aussagen ableiten, die im Folgenden dargelegt werden.

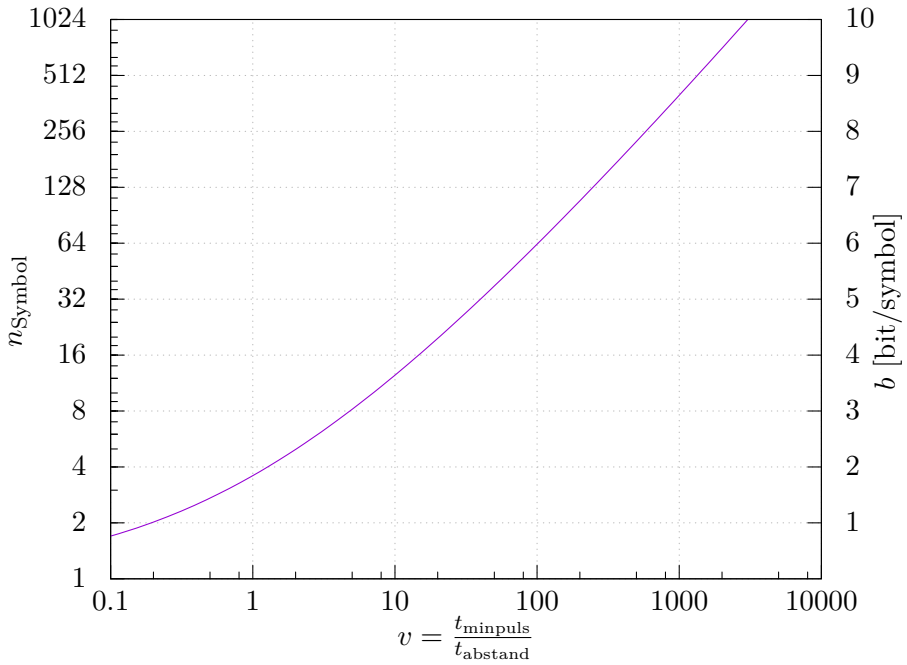

Abbildung 5.4: Die optimale Bit-Anzahl b ist abhängig vom Verhältnis v der minimalen Pulsbreite t_{minpuls} zum Pulsabstand t_{abstand}.

5.4.2 Auswertung der mathematischen Zusammenhänge

Aus Gleichung (5.11) lässt sich ablesen, dass für die optimale Symbolanzahl n_{Symbol} nur das Verhältnis der Zeiten $t_{\text{minpuls}}/t_{\text{abstand}}$ ausschlaggebend ist. Abbildung 5.4 bietet eine grafische Darstellung des Verlaufes des Ergebnisses in Abhängigkeit von $t_{\text{minpuls}}/t_{\text{abstand}}$. Das Verhältnis der Zeiten $v = t_{\text{minpuls}}/t_{\text{abstand}}$ gibt an, wie viel besser die Zeitauflösung der Endgeräte gegenüber der Kanalbandbreite ist. Die optimale Anzahl unterscheidbarer Symbole n_{Symbol} steigt mit besserer Zeitauflösung oder mit sinkender Bandbreite. Die Anzahl der übertragenen Bit je Symbol steigt logarithmisch mit der Anzahl zur Verfügung stehender Symbole.

5.4.3 Spektrale Effizienz

Die absolute Datenrate R ist von beiden Zeiten t_{minpuls} und t_{abstand} abhängig. Um einen Vergleich zur Bit-seriellen Übertragung herstellen zu können, dient die spektrale Effizienz E als Metrik. Als Kanalmodell dient der in Unterabschnitt 2.2.2 ideale Tiefpass. Damit gilt für die Bandbreite B wie bei der Bit-seriellen Übertragung eine minimale Pulsdauer $t_{\text{minpuls}} = 1/(2B)$. Die spektrale Effizienz ist demnach

$$E = \frac{R}{B} = 2t_{\text{minpuls}} \cdot R$$

durch Einsetzen von Gleichung (5.4) folgt

$$E = 2t_{\text{minpuls}} \cdot R = 2t_{\text{minpuls}} \cdot \frac{\log_2 n_{\text{Symbol}}}{2 \cdot t_{\text{minpuls}} + t_{\text{abstand}} \cdot (n_{\text{Symbol}} - 1)}.$$

Nachdem Zähler und Nenner durch t_{minpuls} dividiert werden

$$E = \frac{2 \cdot \log_2 n_{\text{Symbol}}}{2 + \frac{t_{\text{abstand}}}{t_{\text{minpuls}}} \cdot (n_{\text{Symbol}} - 1)},$$

zeigt die so gewonnene Formel, dass die spektrale Effizienz bei optimaler Symbolanzahl nur vom Verhältnis $t_{\text{abstand}}/t_{\text{minpuls}} = 1/v$ abhängig ist, welches genau das Reziproke der Gleichung (5.11) ist. Durch Einsetzen wird die spektrale Effizienz E in Abhängigkeit von v dargestellt:

$$E_{\max} = \frac{\log_2 n_{\text{Symbol}}}{1 + \frac{1}{2} \cdot \frac{n_{\text{Symbol}} - 1}{v}}.$$

Für die optimale Symbolanzahl gilt $\mathrm{d}R/\mathrm{d}n_{\text{Symbol}} = 0$, und damit nach Gleichung 5.11 auch

$$v = \frac{t_{\text{minpuls}}}{t_{\text{abstand}}} = \frac{1}{2} \left(n_{\text{Symbol}} \cdot (\ln n_{\text{Symbol}} - 1) + 1 \right).$$

Durch Einsetzen der Gleichung für v, ergibt sich

$$E_{\max} = \frac{\log_2 n_{\text{Symbol}}}{1 + \frac{1}{2} \cdot \frac{n_{\text{Symbol}} - 1}{\frac{1}{2}\left(n_{\text{Symbol}} \cdot \left(\ln n_{\text{Symbol}} - 1\right) + 1\right)}} = \frac{\log_2 n_{\text{Symbol}}}{1 + \frac{n_{\text{Symbol}} - 1}{n_{\text{Symbol}} \cdot \left(\ln n_{\text{Symbol}} - 1\right) + 1}}$$

Im Nenner erweitert zu

$$E_{\max} = \frac{\log_2 n_{\text{Symbol}}}{\frac{n_{\text{Symbol}} \cdot \left(\ln n_{\text{Symbol}} - 1\right) + 1 + n_{\text{Symbol}} - 1}{n_{\text{Symbol}} \cdot \left(\ln n_{\text{Symbol}} - 1\right) + 1}},$$

ausmultipliziert

$$E_{\max} = \frac{\log_2 n_{\text{Symbol}}}{\frac{n_{\text{Symbol}} \cdot \ln n_{\text{Symbol}} - n_{\text{Symbol}} + 1 + n_{\text{Symbol}} - 1}{n_{\text{Symbol}} \cdot \ln n_{\text{Symbol}} - n_{\text{Symbol}} + 1}}$$

und wieder zusammengefasst

$$E_{\max} = \frac{\log_2 n_{\text{Symbol}}}{\frac{n_{\text{Symbol}} \cdot \ln n_{\text{Symbol}}}{n_{\text{Symbol}} \cdot \ln n_{\text{Symbol}} - n_{\text{Symbol}} + 1}}.$$

Die Multiplikation von Zähler und Nenner mit $(n_{\text{Symbol}} \cdot \ln n_{\text{Symbol}} - n_{\text{Symbol}} + 1)$ ergibt

$$E_{\max} = \frac{\log_2 n_{\text{Symbol}} \cdot (n_{\text{Symbol}} \cdot \ln n_{\text{Symbol}} - n_{\text{Symbol}} + 1)}{n_{\text{Symbol}} \cdot \ln n_{\text{Symbol}}}.$$

Durch ausklammern von $1/\ln 2$ folgt

$$E_{\max} = \frac{1}{\ln 2} \cdot \frac{\ln n_{\text{Symbol}} \cdot (n_{\text{Symbol}} \cdot \ln n_{\text{Symbol}} - n_{\text{Symbol}} + 1)}{n_{\text{Symbol}} \cdot \ln n_{\text{Symbol}}}.$$

Nun wird $\ln n_{\text{Symbol}}$ gekürzt

$$E_{\max} = \frac{1}{\ln 2} \cdot \frac{n_{\text{Symbol}} \cdot \ln n_{\text{Symbol}} - n_{\text{Symbol}} + 1}{n_{\text{Symbol}}}$$

und durch n_{Symbol} dividiert

$$E_{\max} = \frac{1}{\ln 2} \cdot \left(\ln n_{\text{Symbol}} - 1 + \frac{1}{n_{\text{Symbol}}} \right). \tag{5.12}$$

Mit der gewonnen Gleichung (5.12) lässt sich die spektrale Effizienz für die jeweilige optimale Symbolanzahl berechnen.

5.4.4 Abschätzung der Parameter der PWM-Datenübertragung

Die numerische Lösung für Anzahl b Bit pro Symbol von 1 bis 11 ist in Tabelle 5.1 dargestellt. Die Tabelle stellt für jedes b dasjenige Verhältnis $v = t_{\text{minpuls}}/t_{\text{abstand}}$ dar, für das die jeweilige Bit-Anzahl pro Symbol optimal ist. Die verwendeten Formeln sind:

$$v = \frac{1}{2} \cdot (n_{\text{Symbol}} \cdot (\ln n_{\text{Symbol}} - 1) + 1)$$

und

$$E = \frac{1}{\ln 2} \cdot (\ln n_{\text{Symbol}} - 1 + \frac{1}{n_{\text{Symbol}}}).$$

Als Vergleichswert ist die Spalte mit 10 Symbolen eingefügt, die als erste die spektrale Effizienz von 2 Bit/s/Hz übersteigt, welche das theoretische Maximum der spektralen Effizienz der Bit-seriellen Übertragung darstellt. Anhand dessen wird ersichtlich, dass bei $v = t_{\text{minpuls}}/t_{\text{abstand}} > 7$ die PWM-Datenübertragung weniger Bandbreite als die Bit-serielle Übertragung benötigt.

In der Praxis ist als Fragestellung relevant, welche Bandbreite für eine bestimmte festgelegte Datenrate notwendig ist. Abbildung 5.5 beantwortet diese Frage unabhängig von der konkreten Bandbreite, indem sie die spektrale Effizienz E über v darstellt. In der Abbildung sind somit die letzten beiden Spalten der Tabelle 5.1 aufgetragen. Für ein gegebenes Übertragungssystem mit einer Bandbreite B und einer erforderlichen Datenrate R folgt die benötigte spektrale Effizienz $E = R/B$. Aus der Abbildung 5.5 lässt sich nun die Anforderung der Zeitauflösung t_{abstand} ablesen, da $t_{\text{abstand}} = t_{\text{minpuls}}/v = 1/(2B \cdot v)$ gilt. Anhand eines Beispiels soll dies verdeutlicht werden: Beträgt die zur Verfügung stehende Bandbreite 25 MHz, und ist eine Datenrate von 90 MBit/s gefordert, so wird eine spektrale Effizienz von

Tabelle 5.1: Numerische Werte für die optimale Bit-Anzahl b und die damit erzielbare spektrale Effizienz E.

b	n_{Symbol}	$\frac{1}{2} \cdot (n_{\text{Symbol}} \cdot (\ln n_{\text{Symbol}} - 1) + 1)$	$\frac{t_{\text{minpuls}}}{t_{\text{abstand}}}$	E
1	2	$\frac{1}{2} \cdot (2^1 \cdot (\ln 2^1 - 1) + 1)$	0,193	0,279
2	4	$\frac{1}{2} \cdot (2^2 \cdot (\ln 2^2 - 1) + 1)$	1,273	0,918
3	8	$\frac{1}{2} \cdot (2^3 \cdot (\ln 2^3 - 1) + 1)$	4,818	1,738
3,322	10	$\frac{1}{2} \cdot (10 \cdot (\ln 10 - 1) + 1)$	7,013	2,024
4	16	$\frac{1}{2} \cdot (2^4 \cdot (\ln 2^4 - 1) + 1)$	14,68	2,647
5	32	$\frac{1}{2} \cdot (2^5 \cdot (\ln 2^5 - 1) + 1)$	39,95	3,602
6	64	$\frac{1}{2} \cdot (2^6 \cdot (\ln 2^6 - 1) + 1)$	101,6	4,580
7	128	$\frac{1}{2} \cdot (2^7 \cdot (\ln 2^7 - 1) + 1)$	247,0	5,569
8	256	$\frac{1}{2} \cdot (2^8 \cdot (\ln 2^8 - 1) + 1)$	582,3	6,563
9	512	$\frac{1}{2} \cdot (2^9 \cdot (\ln 2^9 - 1) + 1)$	1342	7,560
10	1024	$\frac{1}{2} \cdot (2^{10} \cdot (\ln 2^{10} - 1) + 1)$	3037	8,559
11	2048	$\frac{1}{2} \cdot (2^{11} \cdot (\ln 2^{11} - 1) + 1)$	6784	9,558

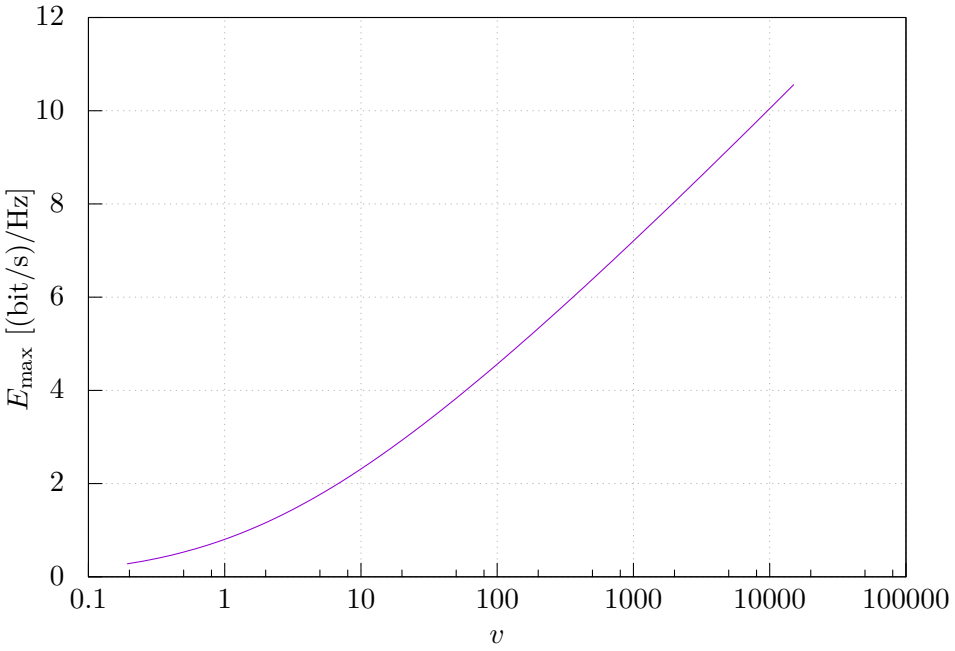

Abbildung 5.5: Die Spektrale Effizienz E über dem Verhältnis v der minimalen Pulsdauer zur Zeitauflösung.

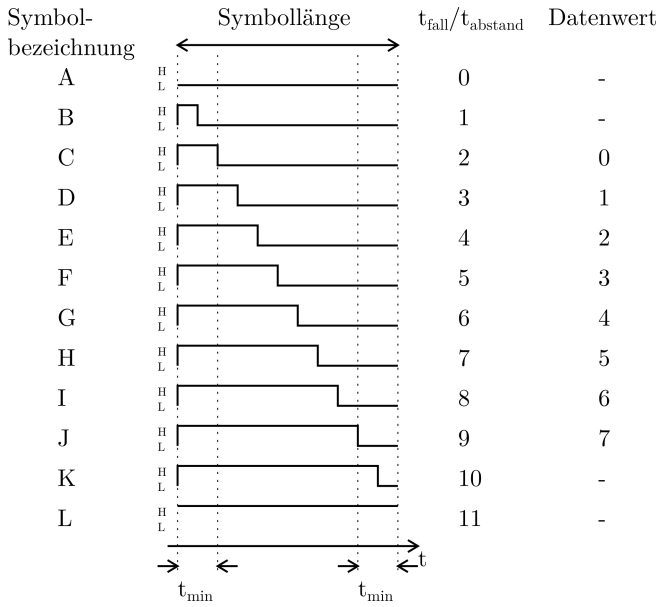

Abbildung 5.6: Symbolauswahl nach der Pulslänge.

$E = 90\,\text{MBit/s}/25\,\text{MHz} = 3{,}6\,\text{Bit/s/Hz}$ benötigt. Laut Abbildung 5.5 muss v zwischen 20 und 50 liegen. Ein Blick in Tabelle 5.1 verrät einen Wert knapp unter 40. Die benötigte Zeitauflösung ist demzufolge $t_{\text{abstand}} = 1/(2 \cdot 25\,\text{MHz} \cdot 40) = 0{,}5\,\text{ns}$.

5.4.5 Anpassung an die vorhandene Bandbreite

Eine wesentliche, neue Idee in dieser Arbeit ist die Trennung der Bandbreitenanforderung von der Zeitauflösung der Pulsweitenmodulation. Durch eine Auswahl der Übertragungssymbole zur Laufzeit kann die Bandbreite der PWM-Datenübertragung auch an unbekannte Kanäle angepasst werden. Die minimale Pulsdauer kann durch Abzählen beliebig gewählt werden. Als Beispiel ist in Abbildung 5.6 die $t_{\text{minpuls}} = 2 \cdot t_{\text{abstand}}$ minimale Pulsdauer doppelt so groß, wie die Zeitauflösung t_{abstand}. Die Symbole sind der Pulslänge nach mit Großbuchstaben bezeichnet. Die acht Symbole C bis J erfüllen diese Vorgabe, und ermöglichen die Übertragung von drei Bit pro Puls. Prinzipiell erfüllen auch die Symbole A und L das Bandbreitenkriterium. Allerdings fehlen diesen beiden Symbolen die Flanken. Es sind also keine Pulse im eigentlichen Sinne. Die Verwendung dieser Symbole ist zwar möglich, erfordert aber Maßnahmen auf höherer Übertragungsprotokollschicht, die eine Synchronisierung der Daten am Empfänger sicherstellen. Da diese höheren Protokollschichten außerhalb des Fokus dieser Arbeit liegen, werden nachfolgend nur Symbole betrachtet, die aus echten Pulsen bestehen.

5.4.6 Der Einfluss schnellerer Hardware

Eine wesentliche, positive Eigenschaft der Pulsweitenmodulation ist, dass durch den Einsatz besserer Sende- und Empfangs-Hardware die Datenrate erhöht werden kann. Die erreichbare Datenrate nimmt dabei logarithmisch mit der Genauigkeit der Zeitmessung und Pulserzeugung zu. Daraus folgt wiederum, das für die Pulsweitenmodulation besonders genaue Zeitmesser und Pulserzeuger benötigt werden. Das folgende Kapitel stellt verschiedene Ansätze zur Zeitmessung und Pulserzeugung vor, und entwirft eine vollständige PWM-Datenübertragungstrecke.

6 FPGA-basierte Implementierung der PWM-Datenübertragung

Die Datenübertragung mittels Pulsweitenmodulation wird für niedrige Datenraten synchron betrieben. Dieses Kapitel stellt Ansätze zur asynchronen Implementierung von Pulserzeugung (Modulator) und Pulsdauermessung (Demodulator) vor. Dabei werden bereits bekannte, asynchrone FPGA-Schaltungen für die PWM-Datenübertragung adaptiert. Auf Basis der asynchronen Signalverarbeitung wird eine unidirektionale Datenübertragungsstrecke entworfen. Darüber hinaus werden Möglichkeiten zur Kalibrierung der Gesamtschaltung besprochen.

Über den Stand der Technik hinaus geht insbesondere die automatische Platzierung der asynchronen Schaltungsteile. Diese Platzierung wird durch die VHDL-Beschreibung des LUT-Inhaltes erreicht. Die Verwendung von in den VHDL-Quelltext eingebetteten Timing-Constraints, ermöglicht die Beschreibung einer asynchronen Schaltung für FPGAs innerhalb einer einzigen VHDL-Datei. Beim gesamten Entwurf liegt das Hauptaugenmerk auf einer einfachen, nachvollziehbaren Implementierung. Die Vermeidung manueller Eingriffe in das Place & Route stellt dafür einen wichtigen Grundstein dar. Dadurch wird sowohl die Portierung auf verschieden große FPGAs der gleichen FPGA-Familie, als auch auf andere FPGA-Familien erleichtert.

Architektur

In Abbildung 6.1 ist die Übertragungsstrecke dargestellt. Sie besteht aus Sender, Kabel, Empfänger und Kalibriereinrichtung. Die Datenübertragung findet folgendermaßen statt: Die Daten werden von der Pulstabelle in eine Pulslänge umgesetzt. Der asynchrone Pulserzeuger gibt den entsprechenden Puls über den FPGA-Pin aus. Die elektrische Verbindung mit dem Empfänger findet über mechanische Adapter und ein Kabel statt. Rechts, auf der Empfängerseite, wird im Pulsmesser die Länge des empfangen Pulses bestimmt. Die Einsentabelle wandelt die gemessene Pulslänge wiederum in den Datenwert um. Die Zuordnung der Pulslänge zum Datenwert wird durch die Kalibrierung bestimmt.

Die asynchrone Pulserzeugung und Pulsmessung beruhen auf der physikalisch gegebenen Signallaufzeit der Verzögerungselemente. Da die Signallaufzeit signifikant von der Temperatur, der Betriebsspannung und den Fertigungstoleranzen des IC

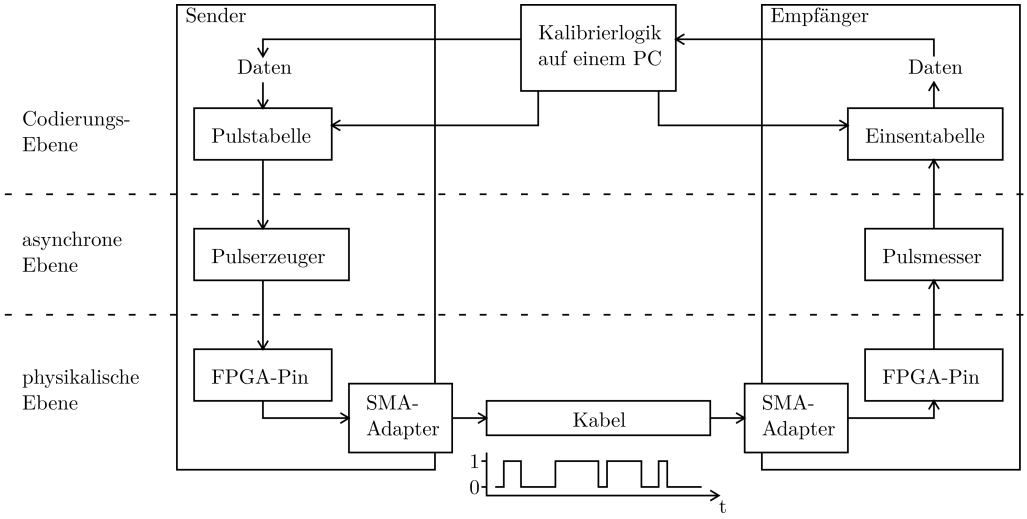

Abbildung 6.1: Die PWM-Datenübertragungsstrecke, links der Sender und rechts der Empfänger. Der Sender erzeugt Pulse variabler Breite anhand der zu übertragenden Daten. Die Pulse werden über das Kabel zum Empfänger übertragen, der durch Messen der Pulslänge auf die Daten schließen kann. Die Kalibrierlogik legt die Tabelleninhalte für eine korrekte Datenübertragung fest.

abhängt, ist ein Vermessen jeder konkreten FPGA-Schaltung unumgänglich. Darüber hinaus erfährt der übertragene Puls auf dem Kabel eine Beeinflussung. Seine Länge ändert sich abhängig von den Kabeleigenschaften. Die genaue Bandbreite des Kabels ist im Allgemeinen unbekannt. Dadurch erfordert eine Ausnutzung der Bandbreite eine Kalibrierungsmessung. Diese Kalibrierung berücksichtigt sowohl die asynchronen Schaltungsteile in Sender und Empfänger als auch die FPGA-Pins und das Kabel mit allen Adaptern.

6.1 Asynchroner Pulserzeuger

Der PWM-Sender hat die Aufgabe, aus einer gewissen Anzahl Datenbits, einen eindeutig zuordenbaren Puls zu erzeugen. Der wesentliche Baustein des Senders ist der Pulserzeuger. Um eine hohe Zeitauflösung zu erreichen, wird der Pulserzeuger mit Hilfe einer asynchronen Verzögerungsleitung realisiert.

In Abbildung 6.2 ist beispielhaft ein Pulserzeuger dargestellt. Er besteht aus einer einstellbaren Verzögerung mit n Stufen, einem Inverter und einem UND-Gatter. Das verzögerte Signal wird invertiert und speist den oberen Eingang des UND-Gatters. Das unverzögerte Signal speist den unteren Eingang des UND-Gatters.

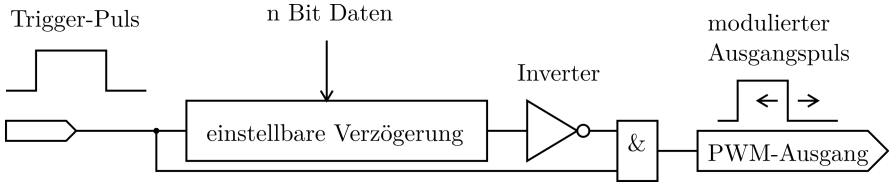

Abbildung 6.2: Struktur eines konfigurierbaren Pulserzeugers.

Funktionsweise

Im Ruhezustand ist das Trigger-Signal Null. Damit liegt der untere Eingang des UND-Gatters auf Null und der andere Eingang hinter dem Inverter auf Eins. Der PWM-Ausgang ist ebenfalls Null. Beim Auftreten der steigenden Flanke des Trigger-Signals liegt an beiden Eingängen des UND-Gatters eine Eins an. Dadurch schaltet der Ausgang ebenfalls auf Eins. Nachdem die steigende Flanke die einstellbare Verzögerungsleitung passiert hat, erzeugt der Inverterausgang eine fallende Flanke. Diese hat am PWM-Ausgang ebenfalls eine fallende Flanke zur Folge. Am Ausgang entsteht somit ein Puls, dessen Länge der Laufzeitdifferenz der beiden UND-Gatter-Zuleitungen entspricht. Die Verwendung einer konfigurierbaren Verzögerungsleitung ermöglicht die Variation der Ausgangspulslänge.

Eigenschaften

Die abgebildete Schaltung ist ein konfigurierbarer Pulserzeuger. Aufgrund des verwendeten UND-Gatters ist der Puls höchstens so lang wie das Trigger-Signal. Die kürzest mögliche Pulsdauer wird durch die Inverter-Laufzeit bestimmt. Die Laufzeit durch die einstellbare Verzögerung bestimmt um welche Zeit der Puls variiert werden kann. Darüber hinaus legt die Wiederholrate des Trigger-Pulses die Pulsrate der PWM fest. Somit sind die Symbollänge und das Modulationsfenster der PWM-Datenübertragung durch Wahl der Schaltungsparameter beeinflussbar.

Verzögerungsleitung

Der Aufbau der einstellbaren Verzögerungsleitung kann mit beliebigen Gattern erfolgen. Da diese Gatter im FPGA immer durch LUTs implementiert werden, sind die Signallaufzeiten durch die LUTs ausschlaggebend für die Größe der Zeitschritte der einstellbaren Verzögerung. Bei Nutzung des Carry-Pfades werden die kleinsten Verzögerungsschritte erreicht. Für die Einstellung eines Verzögerungswertes benötigt die Verzögerungskette Umschaltmöglichkeiten zwischen dem verzögerten und dem unverzögerten Signal. Das Umschalten wird durch Multiplexer

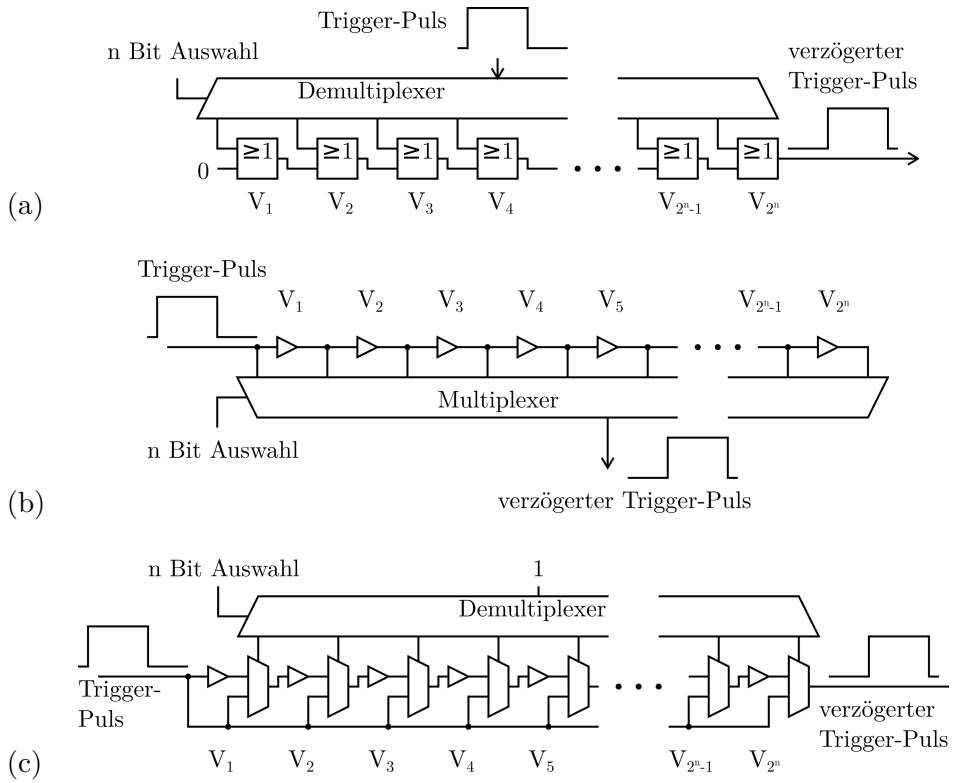

Abbildung 6.3: Drei einstellbare Verzögerungsleitungen: (a) Eingangsdemultiplexer, (b) Ausgangsmultiplexer und (c) Eingangsauswahl.

ermöglicht. Abbildung 6.3 zeigt drei Varianten für einstellbare, asynchrone Verzögerungsketten: (a) den Eingangsdemultiplexer, (b) den Ausgangsmultiplexer und (c) die Eingangsauswahl.

Verzögerungskette mit Eingangsdemultiplexer

Die Schaltung in Abbildung 6.3 (a) besteht aus einem großen Demultiplexer, dessen Ausgänge mit ODER-Gattern verknüpft sind. Der Trigger-Puls wird in den Eingang des Demultiplexers eingespeist. Der n-Bit-Auswahl-Eingang bestimmt, an welchen Demultiplexerausgang der Puls geleitet wird, und damit die Anzahl ODER-Gatter, die der Puls durchlaufen muss. Das Signal muss, bevor es zu den ODER-Gattern gelangt, den Demultiplexer passieren. Dies wirkt sich auf die Verzögerung des Pulses aus.

Verzögerungskette mit Ausgangsmultiplexer

Ein alternativer Aufbau der einstellbaren Verzögerung besteht in einer Verzögerungsleitung, die immer alle Verzögerungselemente durchläuft. Dargestellt ist diese Variante in Abbildung 6.3 (b) Hier verknüpft ein Multiplexer alle Verzögerungselementausgänge. Der n-Bit-Auswahl-Eingang wählt aus, welcher Multiplexereingang aktiv ist. Damit entscheidet er darüber, wie viele Verzögerungselemente passiert werden. In diesem Fall liegt der vollständige Multiplexer im Signalpfad des Trigger-Signals. Dadurch ist ein Umladen der gesamten Schaltung bei jedem Trigger-Puls notwendig. Darüber hinaus wirkt sich die Laufzeit durch den Multiplexerbaum auf die Signallaufzeit aus, welche ohne explizite Platzierung schlecht vorhersagbar ist. Eine explizite Platzierung erschwert wiederum die Parametrierung der Verzögerungselementeanzahl im VHDL-Quelltext, da auch die Platzierungsinformationen an die VHDL-generics angepasst werden müssen.

Verzögerungskette mit Eingangsauswahl

Die dritte Schaltungsvariante in Abbildung 6.3 (c) besteht aus einer Reihe von Verzögerungselementen und Multiplexern. Diese bilden die Verzögerungskette. Die Steuerung der Verzögerungskette erfolgt durch den oben liegenden Demultiplexer, welcher den Auswahlvektor für die Multiplexer erzeugt. Dieser Vektor enthält genau eine Eins an der Position, an der das unverzögerte Signal in die Verzögerungskette eingespeist wird. Der Signaleingang des Demultiplexers liegt fest auf Eins. Die Wahl des Demultiplexerausgangs erfolgt am Auswahleingang mit der Zahl der Verzögerungselemente. Die Größe der Verzögerung ist wiederum von der Anzahl der noch zu passierenden Verzögerungselemente abhängig. Je weiter links die Eins steht, desto größer wird die Verzögerung des Signals. Daher stellt der n-Bit-Auswahleingang des Demultiplexers der Steuereingang für die Anzahl der Verzögerungsstufen dar.

Vergleich der Verzögerungsketten

Sowohl der Eingangsdemultiplexer (a) als auch der Ausgangsmultiplexer (b) liegen direkt im Signalpfad des Trigger-Pulses. Im direkten Vergleich ist das Zeitverhalten der Eingangsauswahl (c) unkritisch. Es besteht sogar die Möglichkeit, den Auswahldemultiplexer bei Bedarf mit zusätzlichen Pipeline-Registern zu versehen, da er nicht im Signalpfad des Trigger-Pulses liegt.

6.2 PWM-Demodulation: Pulsmesser

Auf der Empfangsseite der Datenübertragungsstrecke wird die Länge des Pulses bestimmt. Dabei kommt eine Zeitmessschaltung zum Einsatz, welche die Zeit zwischen der steigenden und der fallenden Flanke des Pulses misst. Die PWM-Datenübertragung stellt dabei einige besondere Anforderungen an die Zeitmessschaltung. Zuerst einmal muss die Zeitmessung eines Pulses schnell beendet sein, damit der nächste Puls empfangen werden kann.

Da die Pulsrate durchaus mehrere 100 MHz betragen kann, bleibt für die eigentliche Messung nur Zeit im einstelligen Nanosekundenbereich. Die Anforderung nach einer möglichst kurzen Messpause leitet sich direkt aus der Pulsform ab. Die fallende Flanke eines Pulses legt das Ende der Zeitmessung fest. Nun bleibt Zeit bis zum Beginn des nächsten Pulses, um die Messung zu verarbeiten. Für den Fall des längsten Pulses entspricht diese Zeit genau der minimalen Pulsdauer, die aufgrund der begrenzten Bandbreite einzuhalten ist. Daraus folgt, dass die vorhandene Bandbreite nur ausgenutzt werden kann, wenn die Messpause kürzer ist, als die minimale Pulsdauer.

Die meisten asynchronen Zeitmessschaltungen werden im Hinblick auf eine hohe Genauigkeit entwickelt. In der Regel geht das zu Lasten anderer Parameter, wie der Messpause. Wu und Shi [WS08] geben für den Wave–Union–TDC eine notwendige Messpause von 45 ns an. Für eine einfacher strukturierte Tapped Delay-Line auf der gleichen Plattform geben sie eine Messpause von 2,5 ns an. Für die PWM-Demodulation kommt daher eine Tapped Delay-Line zum Einsatz. Im Folgenden werden deren Implementierungsmöglichkeiten untersucht.

6.2.1 Tapped Delay-Line

Die schnellste Messwiederholrate, und damit auch die höchste Pulsrate ist mit einer Tapped Delay-Line erreichbar. Die Ursache dafür liegt in der Struktur der FPGA-basierten Tapped Delay-Lines. Ein Ausschnitt einer Tapped Delay-Line ist in Abbildung 6.4 dargestellt.

Aufbau

Der PWM-Demodulator besteht aus einer normalen Tapped Delay-Line, bei der das Startsignal der Eingangspuls ist, und das Stoppsignal der invertierte Eingangspuls ist. Die Inversion erfolgt an den Takteingängen der Register.

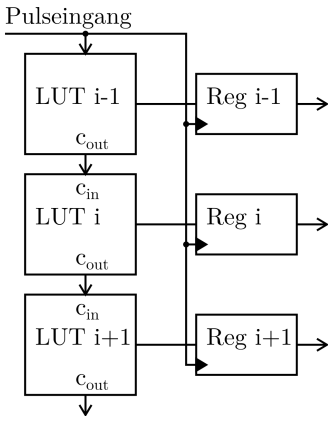

Abbildung 6.4: Tapped Delay-Line zur Pulslängenmessung.

Funktion

Der empfangene Puls wird auf den Carry-Pfad eines LAB geleitet. Die steigende Flanke breitet sich nun nach unten entlang des Carry-Pfades durch die LUTs aus. Dabei werden die Registereingänge `rin(i)` entlang des Pfades auf Eins gesetzt. Sobald die fallende Flanke des Pulses die Register erreicht, wird der Eingangswert übernommen. Die fallende Flanke fungiert so als Stoppsignal des selben Signals, welches auch das Startsignal ist.

VHDL-Beschreibung der Tapped Delay-Line

Die Beschreibung einer Tapped Delay-Line kann auf VHDL erfolgen. Die Stufenanzahl ist generisch, sodass Hardwarebedarf und Messbereich passend zur FPGA-Plattform gewählt werden können. Die VHDL-Implementierung beschreibt direkt den Inhalt der 4-Input-LUT des Cyclone II FPGA:

```
lut: cycloneii_lcell_comb
   generic map (
      lut_mask => "1111000011110000",
      sum_lutc_input => "cin")
   port map (
      cin => cin,
      combout => to_reg,
      cout  => cout );
```

Das am Übertragseingang anliegende Signal `cin` wird entsprechend der `lut_mask` sowohl an den kombinatorischen Ausgang `combout` als auch an den Übertragsausgang `cout` geleitet. Die Verkettung der Verzögerungselemente erfolgt durch die

Verbindung des Übertragsausgangs `cout` eines Elementes (i) mit dem Übertragseingang `cin` des nächsten Elementes durch das VHDL-Signal c(i+1). Im VHDL-Quelltext ist das so realisiert:

```
g: for i in rin'range generate
   p: entity work.pulscarry(lut)
   port map (
      cin   => c(i),
      cout  => c(i+1),
      to_reg=> rin(i) );
end generate g;
```

Die Anzahl der Verzögerungsstufen wird durch die Breite des Signals `rin` bestimmt. Diese recht einfache Beschreibung ist ausreichend, damit der Fitter alle `pulscarry`-Elemente untereinander platziert, und mit dem Carry-Signal verknüpft. Daher kann auf eine explizite Platzierung der Tapped Delay-Line-Elemente verzichtet werden.

Beschreibung des Zeitverhaltens

Um Quartus das Anlegen der offensichtlich langen Leitung durch den Carry-Pfad zu erleichtern, ist es nicht ausreichend durch die Verwendung des Timing-Constraint `set_false_path` die logische Verbindung für den Timing-Analyzer zu kappen. Denn der Fitter sieht trotzdem noch die Länge der Leitung als begrenzend für die Frequenz, und damit als problematisch an. Die Lösung bringt das Timing-Constraint `set_disable_timing`. Dies deaktiviert sämtliche Zeitbetrachtungen des Fitters auf diesem Signal. Das ist insofern hervorzuheben, weil `set_disable_timing` nicht durch die grafischen Timing Analyzer gesetzt werden kann, und auch nicht in der Hilfe dokumentiert wird. Einzig die pdf-Datei der Timing-Analyzer-Referenz [Timequest09] enthält eine Beschreibung dieses Befehls. Das Timing-Constraint für die Tapped Delay-Line lässt sich folgendermaßen angegeben:
`set_disable_timing [get_cells -compatibility_mode {*|r[*]}]`
Wobei `r` der Name des Registersignals ist, das von `rin` gespeist wird.

Eingebettete Timing-Constraints

Für den Ort des Timing-Constraints gibt es nun drei Möglichkeiten:

1. die Angabe in einer Synopsys Design Constraint(sdc)-Datei,
2. die Angabe in der qsf-Datei zusammen mit den anderen Projekteinstellungen oder

3. das Einbetten in ein Altera-spezifisches VHDL-Attribut.

Die erste Variante ist der Standardweg zur Festlegung des Zeitverhaltens einer Schaltung. Die zweite Variante erfordert die Angabe eines `global_assignments` innerhalb der sdc-Datei. Die entsprechende Zeile (zur besseren Lesbarkeit umgebrochen) lautet dann

```
set_global_assignment -name SDC_STATEMENT \
 "set_disable_timing [get_cells -compatibility_mode {*|r[*]}]"
```

Den ersten beiden Varianten ist gemein, dass die Signalnamen, die innerhalb der VHDL-Datei verwendet werden, auch außerhalb der VHDL-Datei in den Einstellungsdateien konsistent gehalten werden müssen. Die dritte Variante vermeidet dieses Problem. Hier wird der Text des `global_assignments` für das Synthese-Werkzeug in eine VHDL-Variable geschrieben. Bei der obigen Tapped Delay-Line sieht das vereinfacht folgendermaßen aus:

```
[...]
architecture arch of tdl is
   constant sdc : string := "-name SDC_STATEMENT " &
      " ""set_disable_timing [get_cells " &
      " -compatibility_mode {*|r[*]}]"" ";
   attribute altera_attribute : string;
   attribute altera_attribute of arch : architecture is sdc;
[...]
```

Das & verkettet VHDL-Zeichenketten, und die doppelten Gänsefüßchen (`""`) ergeben innerhalb der Zeichenkette einfache Gänsefüßchen. Somit entspricht der Inhalt des Attributs `altera_attribute` der Zeichenkette, die in dem obigen sdc-Datei-Ausschnitt nach `set_global_assignment` steht.

6.2.2 Einsenzähler

Der Ergebnisvektor des Zeitmessers ist ein Thermometer-Code, also eine Reihe von Einsen gefolgt von Nullen, etwa so: `11111110000`. Die Anzahl der darin vorhandenen Einsen ist das eigentliche Ergebnis der Zeitmessung. Für die Umsetzung der 2^n Bit des Ergebnisvektors der Tapped Delay-Line auf die n Bit Daten werden die Einsen gezählt. Das Zusammenzählen geschieht mit Addierern. Da die Reihenfolge der einzelnen Additionen keine Rolle spielt, ergeben sich verschiedene Möglichkeiten für die Anordnung der Addierer. Aus der Laufzeit ergibt sich unmittelbar die Pulsrate, die der Einsenzähler verarbeiten kann.

6.3 Symboltabellen

Die Symboltabellen speichern die aus der Kalibrierungsmessung erhaltenen Werte für die Pulslängen. Da die Kalibrierung nach der Programmierung des FPGA stattfindet, müssen die Symboltabellen nachträglich änderbar sein. Daher ist es sinnvoll auf interne RAM-Blöcke des FPGA zurückzugreifen, um nach einer Kalibrierungsmessung die neuen Werte speichern zu können. Der benötigte Speicherplatz liegt bei $n_{\text{Symbol}} \cdot \log_2 n_{\text{Symbol}}$ Bit. Was bei 256 Symbolen zu 0,5 Kibibyte jeweils für Sender und Empfänger führt. Die RAM-Blöcke im Cyclone II FPGA sind 4 KibiByte groß, und damit als Symboltabellen geeignet. Die maximale Taktfrequenz der RAM-Blöcke ist mit 250 MHz angegeben. Für höhere Pulsraten sind andere Lösungen notwendig, die im Detail in Kapitel 9 beschrieben werden.

6.4 Weitere Bestandteile der FPGA-Schaltung

Als Steuersystem für die gesamte Schaltung dient auf jedem FPGA ein Nios II Prozessor (siehe Abschnitt 3.2.2). Dieser verfügt über eine virtuelle serielle Schnittstelle zur Kommunikation mit dem Kalibrier-PC.

Der Standardlogikpegel an den FPGA-Pins ist 3,3 V-LVTTL. Für die PWM-Datenübertragung ist diese Einstellung sinnvoll, um die prinzipielle Leistungsfähigkeit bei Single-Ended-Logic messen zu können. Darüber hinaus bietet das Cyclone II FPGA eine Reihe differentieller Übertragungsstandards. Die Unterdrückung von Gleichtaktstörungen reduziert den Jitter des übertragenen Signals. Daher ist zu erwarten, dass eine differentielle Übertragung weniger Varianz in den Pulslängen aufweist. Sowohl 3,3 V-LVTTL als auch LVDS werden im Rahmen der Arbeit verwendet.

6.5 Zusammenfassung

Dieses Kapitel hat Implementierungsmöglichkeiten für die wesentlichen Schaltungsteile charakterisiert. Es hat Möglichkeiten aufgezeigt, mittels Embedded-SDC-Constraints und FPGA-spezifischen VHDL-Entities die Schaltung vollständig innerhalb ein und derselben VHDL-Datei zu beschreiben. Diese Vorgehensweise hilft, Inkonsistenzen zu vermeiden, und führt zu einfacher parametrierbaren und reproduzierbaren Schaltungen. Das folgende Kapitel beschreibt eine konkrete Umsetzung der PWM-Datenübertragungsstrecke.

7 Versuchsaufbau des Testsystems

Dieses Kapitel beschreibt den Aufbau des ersten Demonstrators für die PWM-Datenübertragung. Das Ziel des Aufbaus besteht im Beweis der prinzipiellen Funktion der asynchronen Pulsweitenmodulation für die Datenübertragung. Das Kapitel besteht aus drei Teilen. Zuerst wird der Gesamtaufbau aus Sender, Empfänger, Kabel und Steuerrechner betrachtet. Im Anschluss wird die FPGA-Konfiguration anhand der einzelnen Komponenten erklärt. Den Abschluss bildet die Vorstellung des Kalibrierverfahrens der Datenübertragungsstrecke.

7.1 Allgemeiner Aufbau

Der Demonstrator in Abbildung 7.1 besteht aus zwei FPGA-Entwicklungsplatinen vom Typ DE2-70 des Herstellers Terasic. Die FPGA-Platinen sind über ein Koaxialkabel von einem Meter Länge miteinander verbunden. Der Anschluss des Koaxialkabels erfolgt mit Adapterplatinen. Die Adapter stellen SMA-Anschlüsse[1]

[1] SubMiniature version A, Koaxialsteckverbinder

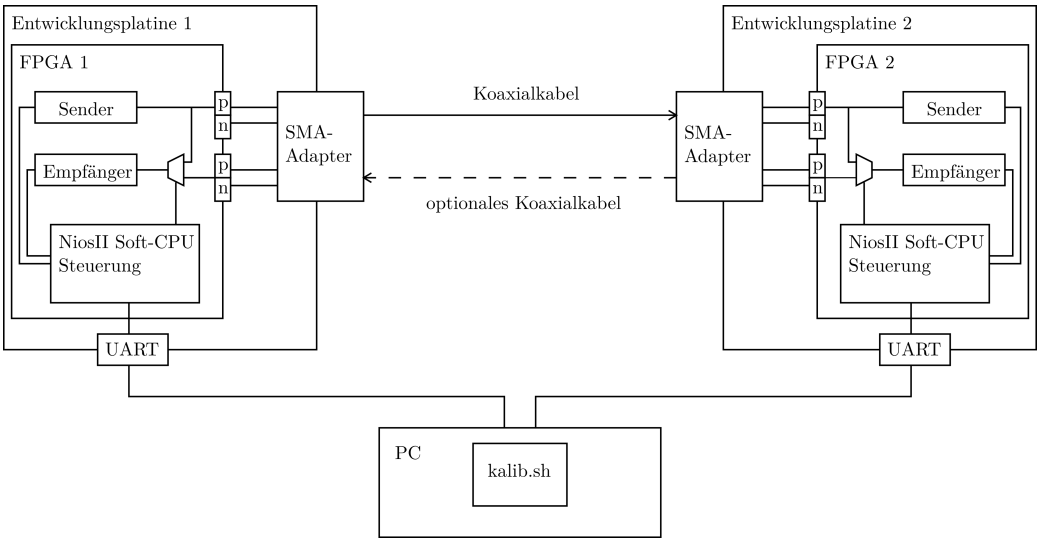

Abbildung 7.1: Der Demonstrator für die PWM-Datenübertragung.

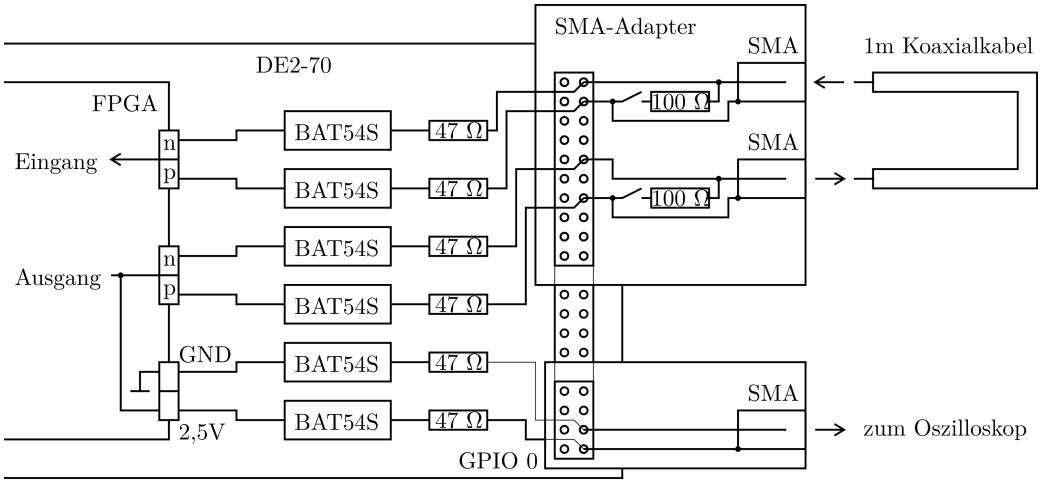

Abbildung 7.2: Aufbau der SMA-Adapterplatine und elektrische Verbindung mit dem FPGA.

an den GPIO-Pins[2] des DE2-70 zur Verfügung. Die Konfiguration der FPGAs und die Steuerung der Messungen erfolgt von einem Rechner aus. Dieser ist per USB-Kabel mit dem USB-Blaster[3] der Entwicklungsplatinen verbunden. Die genaue Funktionsweise der rechnerseitigen Software ist im Rahmen der Beschreibung der Kalibrierung am Ende dieses Kapitels erklärt.

Die elektrische Verbindung von Sender und Empfänger

Als Übertragungskabel dient ein Koaxialkabel vom Typ RG316/U[4] mit 1 m Länge. Das Kabel besitzt an beiden Enden SMA-Stecker. Der Anschluss an die Steckerleisten des DE2-70 erfolgt über selbst gefertigte Adapterplatinen (SMA-Adapter in Abbildung 7.2). Das Kabel kann entweder von einer Entwicklungsplatine zur anderen führen, oder, wie in Abbildung 7.2 dargestellt, die Sende- und Empfangsschaltung derselben Platine miteinander verbinden. Somit sind alle Experimente auch mit einer einzigen Entwicklungsplatine durchführbar.

Anschlussbeschaltung auf dem DE2-70

Auf der Sendeseite dienen die FPGA-Pins `G26` und `G25` als Ausgangs-Pins. Der Pin `G26` ist der p-Pin, und der Pin `G25` ist der n-Pin für das differentielle Signal. Dieses Pin-Paar ist im Datenblatt des Cyclone II FPGA [CycloneII08] mit `LVDS131` bezeichnet. Beide Signalleitungen sind gegen Masse und Betriebsspannung mit

[2] General Purpose I/O
[3] Programmieradapter für Altera-FPGAs
[4] genaue Parameter in Abschnitt 2.2

Schottky-Dioden vom Typ BAT54S abgesichert. Zusätzlich ist ein 47 Ω-Widerstand in Reihe zum Signalpfad geschaltet. Von diesem Widerstand aus, führen Leitungen zur Buchsenleiste GPIO0. Die p-Leitung (IO_A9) führt an Pin 14 und die n-Leitung (IO_A7) an Pin 10.

Am Empfänger dienen an der Buchsenleiste GPIO0, Pin 4 als p-Leitung (IO_A1) und Pin 2 als n-Leitung (IO_A0). Die Leitungen führen ebenfalls über jeweils 47 Ω-Widerstände zum FPGA-Pin-Paar LVDS135, das aus den Pins C29 (p) und C30 (n) besteht.

Für den Signalpegel auf dem Kabel sind zwei Optionen vorgesehen. Einerseits kann der Signalpegel auf 3,3 V-TTL-Pegel eingestellt werden. Dabei wird der p-Pin mit dem Signal beaufschlagt, und der n-Pin FPGA-intern mit GND verbunden. Dadurch ist eine einfache Beobachtung der Signale mit dem Oszilloskop möglich. Andererseits ist durch die Verwendung eines Pin-Paares auch ein differentieller Spannungspegel nach dem LVDS-Standard möglich. Um dabei weiterhin die Signalform beobachten zu können, wird das Sendesignal zusätzlich über das FPGA-Pin-Paar LVDS160 mit den Pins P30 und P29 zu den GPIO0-Pins 40 und 38 geführt. Dabei bildet P29 das Bezugspotential durch eine FPGA-interne Verbindung zu GND. P30 ist auf den I/O-Standard 2,5 V-CMOS eingestellt.

Adapterplatinen

Der SMA-Adapter (siehe Abbildung 7.2) ist folgendermaßen aufgebaut: Die SMA-Buchsen sind auf eine Lochrasterplatine aufgelötet. Als Abschlusswiderstand dient ein 100 Ω-Widerstand der über eine Steckbrücke das Gehäuse und den Mittelleiter der SMA-Buchse verbindet. Die 100 Ω ergeben sich, weil im FPGA selbst bei der Verwendung des LVDS-Standards automatisch ein 100 Ω-Widerstand zugeschaltet wird. Der Wellenwiderstand des Koaxialkabels beträgt 50 Ω, was zu Signalreflexionen führen würde. Diese werden durch den zusätzlichen, mittels Steckbrücke schaltbaren 100 Ω-Widerstand verhindert. Die Steckbrücke ist auf der Empfängerseite geschlossen. Jede SMA-Buchse ist mit einem Pin-Paar des FPGAs verbunden, der p-Pin mit dem Mittelleiter, und der n-Pin mit dem Gehäuse der SMA-Buchse.

7.2 FPGA-Schaltung des ersten Prototyps

Beide Entwicklungsplatinen besitzen je einen Cyclone II FPGA EP2C70F896C6 mit 68416 Logikelementen. Beide FPGAs werden mit dem selben Bit-Stream konfiguriert. Der Bit-Stream enthält Sender, Empfänger und Nios II-Prozessor. In Abbildung 7.3 ist die Anordnung der Logikelemente im FPGA dargestellt. Auf der linken Seite befindet sich der Sender. In der Mitte ist der Nios II-Prozessor

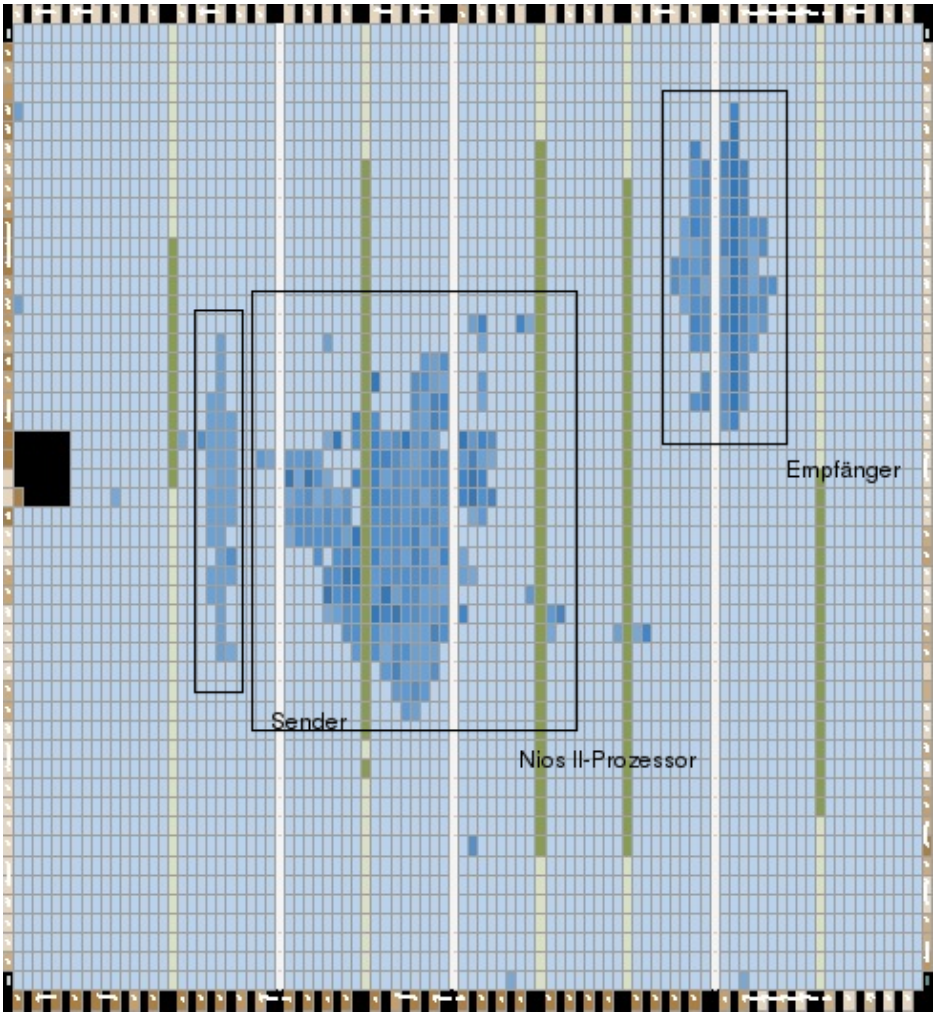

Abbildung 7.3: Der FPGA-Floorplan der pwm-a-Schaltung.

angeordnet. Rechts oben befindet sich der Empfänger. Neben der normalen Datenübertragung zwischen den beiden FPGA-Platinen ist auch eine FPGA-interne Verbindung von Sender- und Empfängerschaltung auswählbar. Dabei verlässt das PWM-Signal das FPGA nicht. Damit ist ein Funktionstest ohne Kabeleinflüsse möglich.

Der Sender

Der Pulsgenerator besteht im Wesentlichen aus der einstellbaren Verzögerungskette. Die Verzögerungskette weist 256 Stufen auf. Damit sind Pulse bis zu einer Breite von etwa $256 \cdot 130\,\text{ps} = 33\,\text{ns}$ erzeugbar. Die einzelnen Verzögerungselemente sind als VHDL-Entity `cycloneii_lcell_comb` implementiert. Dadurch ist

es möglich, auf eine explizite FPGA-spezifische Platzierung zu verzichten. Die Pulsrate beträgt 10 MHz. Diese niedrige Frequenz erleichtert Quartus das Einhalten der Zeitvorgaben beim Verbinden der Logikelemente. Dadurch erhöht sich die Wahrscheinlichkeit, dass die fertig synthetisierte Schaltung auch funktioniert. Die asynchrone Pulserzeugung dieser ersten Schaltungsvariante (im weiteren Verlauf als Variante A oder pwm-a bezeichnet) verwendet 553 LUTs und 1 Register.

Der Empfänger

Der Empfängerzeitmesser ist eine einfache Tapped Delay-Line mit 256 Stufen, also ebenso vielen, wie in der Verzögerungsleitung des Senders. Dieser Teil des Empfängers benötigt 769 Logikelemente. Das sind 3 Logikelemente je Verzögerungsstufe.

Der nächste Baustein des Empfängers ist der Einsenzähler. Zum Test des Synthesewerkzeuges wird nicht festgelegt, wie die Addierer angeordnet werden sollen, sondern das Verhalten der Schaltung wird im VHDL-Quelltext beschrieben:

```
function einsen_zaehlen ( v : std_logic_vector )
return natural is
   variable anzahl : natural := 0;
begin
   for i in v'range loop
      if v(i) = '1' then
         anzahl := anzahl + 1;
      end if;
   end loop;
   return anzahl;
end einsen_zaehlen;
```

Die Beschreibung entspricht einer Addiererkette. Jedoch ist das Synthesewerkzeug frei, anhand äquivalenter Umformungen andere Strukturen zu erzeugen. Aufgrund der niedrigen Pulsrate von 10 MHz ist das Zeitverhalten unkritisch. Aus obiger Beschreibung erzeugt das Synthesewerkzeug 517 Logikelemente, die 255 kaskadierte Addierer implementieren.

Die Symboltabellen

Die Symboltabellen sind als Dual-Port-RAM mit M4K-RAM-Blöcken implementiert. Dadurch können sie einerseits im laufenden Betrieb durch den Nios II-Prozessor mit Werten beschrieben werden, andererseits bleibt die Pulsrate unabhängig vom Takt des Prozessors.

Die Größe einer Symboltabelle ergibt sich aus der Anzahl der Eingangs- und Ausgangswerte. Für den Pulsgenerator, mit seinen 256 Verzögerungsstufen, wird ein

8 Bit-Ausgang benötigt, daher sind die Speicherzellen auf 1 Byte Breite konfiguriert. Der Adresseingang ist ebenfalls 8 Bit breit, und kann demzufolge 256 Adressen ansprechen. Von den 4 KiByte werden also nur 256 Byte genutzt. Die Größe der Empfängersymboltabelle beträgt ebenfalls $256 \cdot 8$ Bit. Der Inhalt der RAM-Blöcke kann sowohl zur Synthese definiert, als auch später über das Programm des Nios II-Prozessors beliebig verändert werden.

Der Nios II-Prozessor

Der Nios II-Prozessor übernimmt die Steuerung der Sende- und Empfangsschaltung. Darüber hinaus verfügt der Prozessor über eine JTAG-UART. Das ist eine virtuelle serielle Schnittstelle, zur Kommunikation mit dem Steuerrechner. Auf dem Nios II-Prozessor läuft eine in C++ geschriebene Eingabeaufforderung. Diese wird vom PC aus mit dem Programm `nios-terminal` aus dem Quartus-Programmpaket gesteuert. Über die Eingabeaufforderung lassen sich Daten senden und die Symboltabellen lesen und schreiben.

Die Latenz der Übertragung

Eine Datenübertragung findet nun folgendermaßen statt: Der Datenwert wird beim Sender an den Adresseingang des RAM-Blockes für die Symboltabelle angelegt. Im nächsten Takt nach 100 ns steht die Konfiguration der Verzögerungsstrecke am Ausgang des RAM-Blocks zur Verfügung. Die Multiplexer der Verzögerungskette schalten den entsprechenden Signalpfad durch. Im darauf folgenden Takt ($t = 200$ ns) löst das Trigger-Signal am Pulserzeuger das Senden eines Pulses aus. Das Signal benötigt im Extremfall etwa $256 \cdot 130$ ps ≈ 35 ns zum Durchlaufen der Pulserzeugerschaltung. Der so erzeugte Puls erreicht in etwa 2 ns den Ausgangspin des FPGAs. Die Laufzeit durch den Ausgangspin wird im Datenblatt als „I/O Output Delay“ [CycloneII08, Seite 5-42, Tabelle 5-42] bezeichnet, und beträgt rund 2 ns. Das heißt, dass der Puls das Sende-FPGA 239 ns nach dem Einstellen des Datenwertes verlässt. Die Leiterbahnen auf der Entwicklungsplatine und die Adapterplatinen tragen aufgrund der Serien-Widerstände und des sich daraus ergebenden Tiefpassverhaltens ebenfalls etwa 5 ns zur Signallaufzeit bei. Die Signallaufzeit des Koaxialkabels liegt bei 5 ns. Zum Zeitpunkt $t = 254$ ns erreicht der Puls das Empfangs-FPGA. Die interne Verzögerung des FPGA-Eingangs-Pins („I/O Input Delay“) [CycloneII08, Seite 5-34, Tabelle 5-40]. beträgt etwa 1 ns. Durch die FPGA-internen Leitungen beginnt etwa 2 ns später, die steigende Flanke durch die Verzögerungselemente der Tapped Delay-Line zu „laufen“. Nach Ablauf der Pulsdauer, also spätestens nach weiteren 35 ns liegt die fallende Flanke an den Takteingängen der Tapped Delay-Line-Register an. Die Register übernehmen ihre Eingangswerte. Etwa zum Zeitpunkt $t = 292$ ns liegt der Thermometer-Code am Tapped Delay-Line-Ausgang an. Der Empfängertakt kann prinzipiell in jeder

beliebigen Phasenlage zum Sender stehen. Im schlechtesten Fall wird der neue Thermometer-Code erst 100 ns später (t=392 ns) in den Einsenzähler übernommen. Das Ausgangsregister des Einsenzählers enthält bei $t = 492$ ns die erkannte Pulslänge. Mit diesem Wert am Adresseingang der Empfängersymboltabelle steht nach einem weiteren Takt, und insgesamt $t = 592$ ns der Datenwert im Empfänger zur Verfügung.

Syntheseergebnis

Die Größe der gesamten FPGA-Schaltung beträgt 4900 Logikelemente (LE) bei 256-stufigem Sender und Empfänger. Die Schaltung belegt 7 % des FPGAs, die sich folgendermaßen aufteilen: Der Sender benötigt 750 Logikelemente, und der Empfänger benötigt 1350 Logikelemente. Das entspricht 3 % der FPGA-Ressourcen. Die für die Steuerung eingesetzte Nios II-Prozessor ist ähnlich groß, und benötigt mit 2800 LE etwa 4 % der FPGA-Ressourcen. Als Taktgeschwindigkeit für den digitalen Schaltungsteil werden 50 MHz verwendet, da erfahrungsgemäß der Nios II dann unproblematisch synthetisierbar ist. Eine PLL erzeugt den 50 MHz Nios II-Takt, und das Trigger-Signal. Dadurch ist die Pulsrate zur Synthesezeit wählbar.

Geometrische Anordnung

Die in Abbildung 7.3 dargestellte Anordnung der Logikelemente ist automatisch vom Synthesewerkzeug so gewählt worden. Die komplette FPGA-Schaltung verwendet keine explizite Platzierung. Damit ist die Anordnung zufällig [Ngu16, Folie 20]. Gut erkennbar ist jedoch bei Sende- und Empfangsschaltung jeweils die Spalte von 16 LABs, die jeweils die 256 Verzögerungselemente des Carry-Pfades aufnehmen. Die Vermeidung der expliziten Platzierung bringt den Vorteil der Anwendbarkeit der Schaltung auf allen FPGAs der Cyclone II FPGA-Familie und darüber hinaus bis hin zum Cyclone IV mit sich.

7.3 Kalibriermethode des Gesamtsystems

Für die Kalibrierung ist eine bestehende Kommunikationsmöglichkeit zwischen Sender und Empfänger erforderlich. Die Kommunikation erfolgt über die seriellen Verbindungen mit dem Steuerrechner und die Kommandozeile auf dem jeweiligen Nios II-Prozessor. Der Steuerrechner stellt die korrekte Zuordnung von den gesendeten Daten des Senders zu empfangenen Daten des Empfängers sicher. Das ist notwendig, weil im Empfänger zwei Registerstufen von der fallenden Pulsflanke getaktet werden. Daher steht der erste Datenwert am Empfänger erst nach dem Senden von drei Pulsen zur Verfügung.

Die Kalibrierung beinhaltet das Senden aller möglichen Sendesymbole. Das sind alle 256 wählbaren Verzögerungsstufen des Senders. Alle empfangenen Pulse wiederum werden nach ihrem Thermometer-Code sortiert. Aus Pulsen mit identischen Thermometer-Codes wird einer ausgewählt, so dass im Sender möglichst wenig aktive Elemente an der Pulserzeugung beteiligt sind. So lässt sich eine Tabelle gewinnen, die jedem Thermometer-Code des Empfängers einen Eingangswert am Sender zuordnet. Jede Tabellenzeile stellt dabei einen unterscheidbaren Datenwert dar.

Ablauf der Zuordnung der Tabelleninhalte

Zuerst wird der FPGA-Bit-Stream geladen, und das Nios II-Programm gestartet. Die Symboltabellen werden mit der jeweiligen Adresse als Speicherinhalt gefüllt. Danach haben beide Symboltabellen folgenden Inhalt:

Adresse	Speicherinhalt
0	0
1	1
2	2
3	3
...	...

Somit geben die Symboltabellen den gleichen Wert aus, der am Adresseingang anliegt. Sie verhalten sich transparent, als ob sie nicht vorhanden wären. Die Kalibrierungsmessung kann nun das Verhalten der asynchronen Schaltungsteile und des Kabels bestimmen, die auf dem Signalpfad zwischen den Symboltabellen liegen. Dazu sendet ein Skript 256 Pulse und speichert die Empfangswerte. Das Ergebnis ist eine Tabelle von Sende- und Empfangspulsen, beispielsweise:

Sendepuls	Empfangspuls
...	...
104	83
105	84
106	85
107	86
...	...

Diese Messung erfolgt mehrfach, um die Einflüsse von Temperaturänderungen und zufälligen Fehlern abschätzen zu können. Diejenigen Sendepulse, die reproduzierbar vom Empfänger erkannt werden, können dann als Datensymbole verwendet werden. Nicht eindeutig erkennbare Sendepulse werden verworfen. Den übriggebliebenen Pulsen werden nun der Reihe nach Datenwerte zugeordnet, wie in der folgende Tabelle gezeigt ist:

Sendepuls	Empfangspuls	Datenwert
...	...	...
104	83	61
105	84	62
106	85	63
107	86	64
...	...	...

Im Anschluss werden die Symboltabellen in Sender und Empfänger mit den so gewonnenen Daten gefüllt. Dies ist in Abbildung 7.4 dargestellt. Im Sender ist die Spalte „Datenwert“ die Speicheradresse, und die Spalte „Sendepuls der Speicherinhalt“. Im Empfänger ist die Spalte „Empfangspuls“ die Adresse, und die Spalte Datenwert der Speicherinhalt.

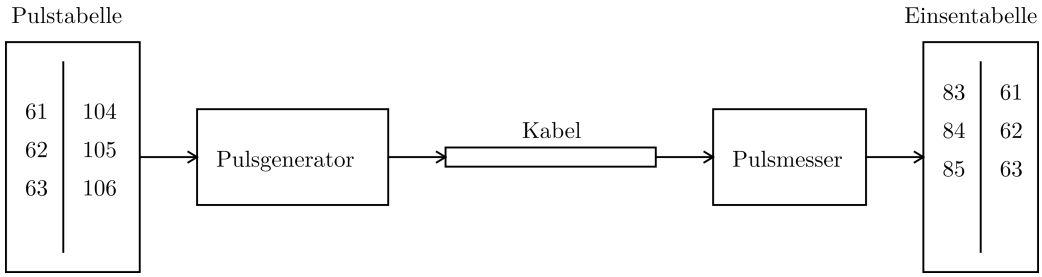

Abbildung 7.4: Aus der Kalibrierung wird der Inhalt von Pulstabelle und Einsentabelle bestimmt.

8 Ergebnisse

Dieses Kapitel stellt erste Ergebnisse der umgesetzten PWM-Datenübertragung dar, und zeigt damit deren prinzipielle Funktion. Die Datenübertragungsstrecke erreicht eine Datenrate von 70 MBit/s bei einer Latenz von 600 ns (6 Takte bei 10 MHz). Weiterhin wird der Temperatureinfluss und der Energiebedarf abgeschätzt.

8.1 Welche Pulse werden erkannt?

Abbildung 8.1 stellt die Empfangswerte von allen 256 möglichen Sendepulsen dar. Deutlich erkennbar ist der annähernd lineare Verlauf zwischen Sendesymbol 32

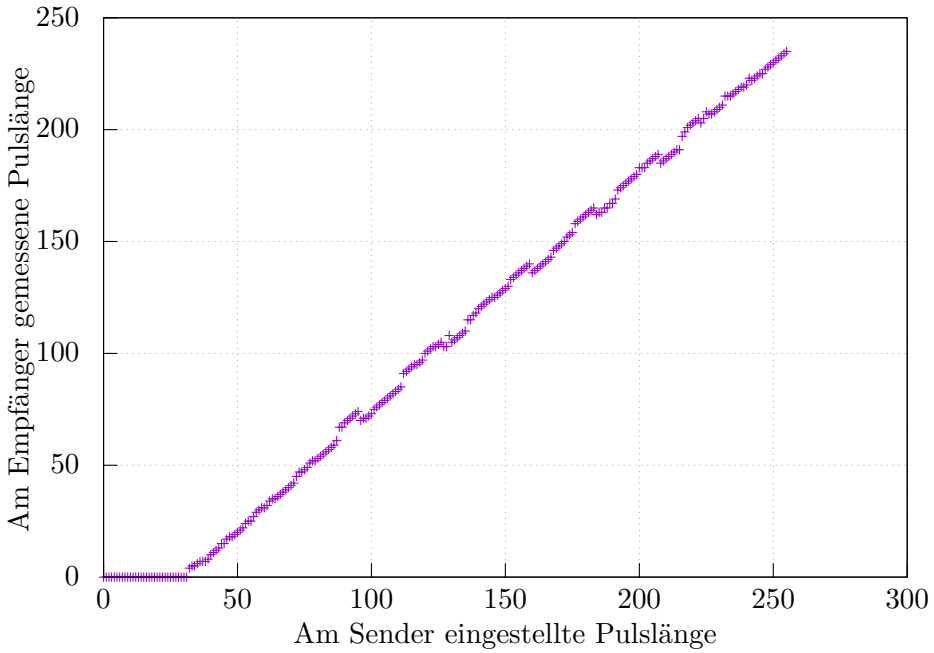

Abbildung 8.1: Eingestellte Pulslängen am Sender und dazugehörige, gemessene Pulslängen im Empfänger, jeweils in Verzögerungschritten zu rund 130 ps.

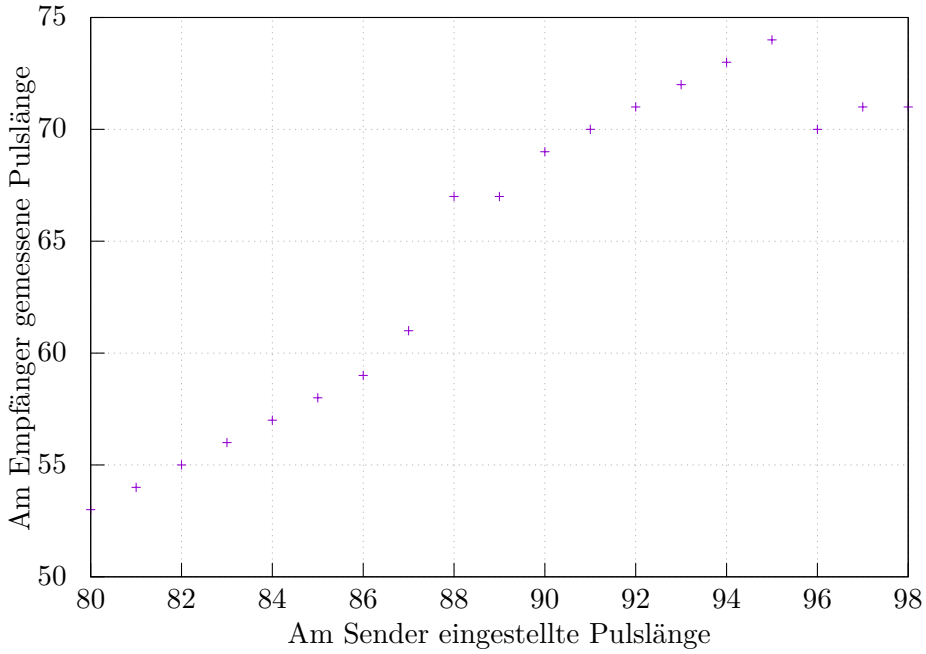

Abbildung 8.2: Ausschnitt aus den empfangenen Pulslängen.

und Sendesymbol 255. Weiterhin ist erkennbar, dass die kürzesten Pulse, bis zum Sendesymbol 31, am Empfänger nicht detektiert werden. Sie können daher nicht für die Datenübertragung verwendet werden.

Am Ausschnitt in Abbildung 8.2 ist erkennbar, dass der Verlauf nicht stetig ist. Zwischen den Sendesymbolen 87 und 88 springt der Empfangswert von 61 auf 67 nach oben. Das nachfolgende Symbol 89 hat den selben Empfangswert 67. Und zwischen Symbol 95 und 96 springt der Empfangswert nach unten. Auch hier wiederholt sich der Empfangswert 70, den auch das Sendesymbol 91 aufweist.

Die Unstetigkeiten nach jeweils 8 Elementen spiegeln die interne LAB-Struktur des FPGA wieder. Die Länge und Anordnung der lokalen Zuleitungen innerhalb des LAB zu den einzelnen LUTs spielt hierbei eine Rolle. Da die exakte Leitungsführung innerhalb des FPGA ein Geschäftsgeheimnis von Altera ist, ist über diese Feststellung hinaus momentan keine weitergehende Begründung möglich.

Für die Symbolauswahl sind die Sprünge und die benachbarten identischen Empfangswerte relevant, da keine eindeutige Zuordnung der Pulse am Empfänger möglich ist. Im Rahmen der Kalibrierung werden daher eindeutige Sendesymbole ausgewählt. Dadurch reduziert sich die Anzahl der nutzbaren Symbole, gegenüber der Anzahl der Verzögerungsstufen.

8.1.1 Zuordnung der Datenwerte

Die Kalibrierung besteht im ersten Schritt auf dem in Abbildung 8.1 dargestellten Durchmessen der Schaltung. Die Ergebnisse bilden die Grundlage für das Füllen der sender- und empfängerseitigen Pulstabellen. Der in Abschnitt 7.3 vorgestellte Algorithmus ordnet von links nach rechts jeder eindeutig erkennbaren Kombination aus Sende- und Empfangssymbol einen Datenwert zu. Für die in Abbildung 8.1 dargestellte Messung ergeben sich folgende Werte: Das Sendesymbol 33, das beim Empfänger als 5 erkannt wird, erhält den Datenwert 1. Das letzte Symbol enthält dann den Datenwert 172. Die so gewonnene Zuordnung von Sendesymbol zu Datenwert wird für die Symboltabelle des Senders verwendet. Die Zuordnung der Datenwerte zum Empfangssymbol dient im Empfänger zur Decodierung.

8.1.2 Datenrate und Latenz

Für die Nutzung zur Datenübertragung verbleiben 172 vom Empfänger unterscheidbare Zustände. Aus diesen 172 Pulsen können nun 128 Symbole beliebig gewählt werden, um sieben Bit pro Puls zu übertragen. Die eingestellte Pulsrate bei der FPGA-Synthese betrug 10 MHz. Die theoretisch erreichbare Datenrate des Aufbaus liegt demzufolge bei $R = 10^7\,\text{Pulse/s} \cdot 7\,\text{Bit/Puls} = 70\,\text{MBit/s}$. Die vorliegende Schaltung ist in der Lage, bei einer Pulsrate von 10 MHz sieben Bit pro Puls mit einer Verzögerung von etwa 600 ns zu übertragen.

Im Hinblick auf die theoretische Betrachtung in Abschnitt 5.3 fällt auf, dass diese Datenrate das Potential der Schaltung nur zu einem kleinen Bruchteil nutzt. Die Schrittweite t_{abstand} der PWM beträgt aufgrund der Struktur der Verzögerungsketten im Mittel 130 ps. Die zur Verfügung stehende Bandbreite lässt sich anhand des kürzesten empfangenen Pulses abschätzen: Das Sendesymbol 33 erzeugt in der Empfänger-Tapped Delay-Line fünf Einsen. Daher muss dieser Puls am Empfänger mindestens eine Länge von $t_{\text{minpuls}} = 5 \cdot 130\,\text{ps} = 650\,\text{ps}$ haben. Daraus ergäbe sich eine Bandbreite von $B = 1/(2 \cdot 650\,\text{ps}) = 769\,\text{MHz}$, die mehr als doppelt so groß ist, wie die mit 400 MHz spezifizierte Bandbreite der Eingangspins des FPGA [CycloneII08, Seite 5-48, Tabelle 5-44]. Einerseits wären bei 10 MHz Pulsrate somit $n_{\text{Symbol}} = (100\,\text{ns} - 2 \cdot 0{,}65\,\text{ns})/130\,\text{ps} = 759$ verschiedene Symbole möglich, was die verwendeten Verzögerungsketten gar nicht abdecken können. Andererseits würde, aufgrund des kleinen Verhältnisses $t_{\text{minpuls}}/t_{\text{abstand}}$, die optimale Datenrate erst bei viel höheren Pulsraten erreicht. Im Detail geht Kapitel 9 darauf ein. Es stellt einen Aufbau vor, der höhere Pulsraten ermöglicht, und der prinzipiell in der Lage ist, 1 GBit/s zu übertragen.

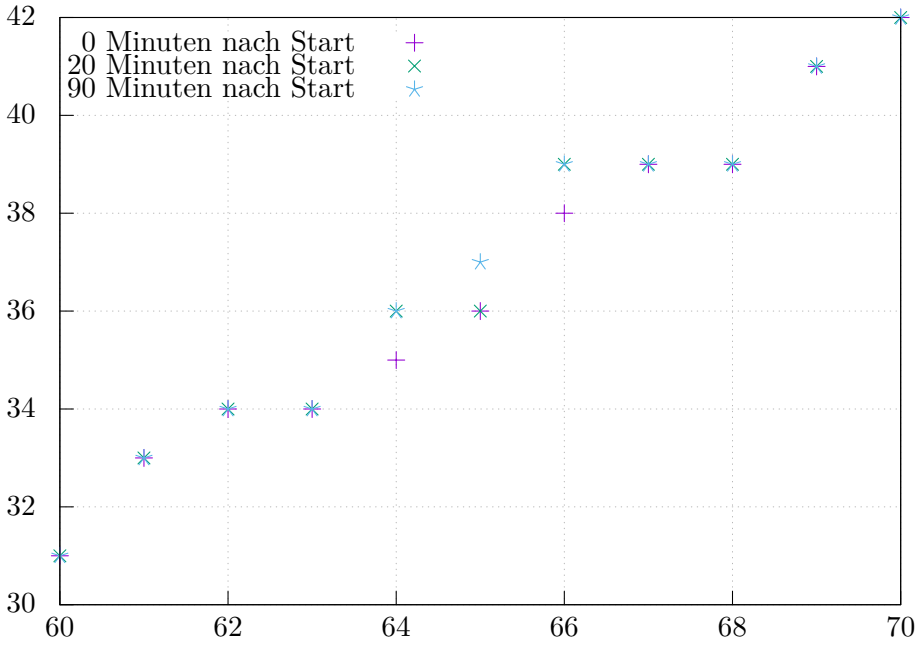

Abbildung 8.3: Unterschiedliche gemessene Pulslängen aufgrund der Erwärmung über die Betriebsdauer.

8.2 Temperatureinfluss

Einen entscheidenden Einfluss darauf, dass die theoretische Datenrate nicht erreicht wird, hat die Temperatur. Wiederholte Messungen der Empfangspulse führen zu leicht abweichenden Ergebnissen. Die Hauptursache dafür ist die Erwärmung der Schaltung während des Betriebes. Die Kurve in Abbildung 8.1 gilt also nur für die Temperatur, unter der sie gemessen wurde. Für den praktischen Betrieb bedeutet das, dass erst nach Erreichen eines Wärmegleichgewichts mit entsprechender Konstanz der Temperatur längere Messreihen möglich sind. Die Zeit für das „Warmlaufen“ liegt im zweistelligen Minutenbereich. Selbst nach dem Warmlaufen hängt die Schaltungstemperatur von der veränderlichen Umgebungstemperatur ab. Abbildung 8.3 zeigt einen Ausschnitt aus den Werten für drei verschiedene Betriebsdauern: direkt nach dem Anschalten, nach 20 Minuten und nach 90 Minuten. Dort ist erkennbar, dass nur einige Sendepulse temperaturabhängigen Schwankungen unterworfen sind. Hier sind die Pulse 64, 65 und 66 betroffen. Vermutlich liegen diese Pulse sehr nah an der Grenze zur nächsten Verzögerungsstufe, sodass eine Veränderung aller Pulslängen um z.B. 10 ps nur bei diesen Pulsen ein anderes Ergebnis auslöst. So wird der Sendepuls 64 nach 0 Minuten als Empfangspuls 35 empfangen. Nach 20 Minuten und nach 90 Minuten ist der Empfangspuls 36 Elemente lang. Dieser Effekt betrifft ebenfalls den Sendepuls 65. Dieser zeigt initial den Empfangswert 36. Nach 90 Minuten Schaltungsbetrieb

empfängt man jedoch eine 37. Auffällig ist, dass sich die Werte immer zu größeren Werten hin verschieben.

8.3 Energiebedarf

Die Abschätzung des Quartus PowerPlay Power Analyzer gibt einen Überblick über den Energiebedarf der gesamten FPGA-Schaltung. Dabei entfallen auf die Nios II-CPU 177 mW. Die Tapped Delay-Line benötigt 3 mW, der Einsenzähler ebenfalls. Der Pulsgenerator erfordert 3 mW. Die RAM-Blöcke der Symboltabellen werden mit jeweils etwa 10 mW angegeben. Leider ist die blockweise Angabe der Leistung an die Hierarchie der VHDL-Quelldateien gekoppelt. Dadurch lassen sich die Funktionsgruppen der Top-Level-Entity nicht separieren. Als Energiebedarf der Top-Level-Entity werden 433 mW angegeben. Das ist mehr, als alle anderen Komponenten zusammen benötigen. Die Ursache ist, dass insbesondere der Energiebedarf der FPGA-globalen Leitungen und der Ein- und Ausgangs-Pins zur Top-Level-Entity gerechnet werden. Die angegebenen Werte für die Komponenten sind also eher als qualitative Größen zu betrachten. Für einen direkten Vergleich mit anderen Übertragungssystemen müsste die Implementierung im selben FPGA Bit-Stream erfolgen, um identische Randbedingungen zu erreichen. Für den qualitativen Vergleich sind diese Werte aber nichtsdestotrotz brauchbar. Ein Vergleich mit den nachfolgend untersuchten Schaltungsvarianten erfolgt in Abschnitt 10.4.2.

8.4 Bewertung des Machbarkeitsnachweises

Zusammenfassend zeigen die Ergebnisse, dass der vorgestellte Aufbau etwa 170 unterscheidbare Pulslängen erzeugt, die für eine Datenübertragung nutzbar sind. Die Kodierung von sieben Datenbit je Puls führt zu einer Datenrate von 70 MBit/s bei einer Pulsrate von 10 MHz. Damit zeigen die Ergebnisse die prinzipielle Machbarkeit der PWM-Datenübertragung mit asynchroner Logik. Allerdings zeigt der vorgestellte Aufbau eine Reihe von Limitierungen.

8.4.1 Temperaturabhängigkeit und Fehlerkorrektur

Die Datenrate von 70 MBit/s wird nur erreicht, wenn keine Übertragungsfehler auftreten. Aufgrund der asynchronen Schaltungen werden Temperaturschwankungen jedoch zu Fehlern in der Übertragung führen. Dies ließe sich durch eine Berücksichtigung verschiedener Temperaturen bei der Kalibrierung verhindern. Auf höherer Protokollebene ließe sich durch das Einfügen von Redundanz, bei gleichzeitiger Reduzierung der Datenrate eine gewisse Fehlertoleranz erreichen. In der

einfachsten Ausprägung, wird nur jeder zweite Datenwert genutzt. Damit wird ein Bit weniger pro Puls übertragen. Tritt jetzt eine Erwärmung der Schaltung auf, so kann der Empfänger den längeren empfangenen Puls dem richtigen Datenwert zuordnen. Somit wäre für kleine Temperaturänderungen die Übertragung fehlerfrei möglich. Aufwendigere Kodierungsverfahren zur Vorwärtsfehlerkorrektur können bei geringerem Overhead eine größere Fehlertoleranz erreichen. An diesem Punkt stellt sich auch die Frage, bei welcher Fehlerrate eine Neukalibrierung sinnvoll ist. Die Implementierung der höheren Protokollschichten und die Beantwortung der dabei auftretenden Forschungfragen liegt außerhalb des Fokus dieser Arbeit, und ist der weiterführenden Forschung vorbehalten.

8.4.2 Verbesserungen der asynchronen Schaltungen

Die Tapped Delay-Line erreicht auf dem Cyclone II FPGA eine gute Zeitauflösung von etwa 130 ps. Die besten FPGA-basierten Zeitmesser erreichen Zeitauflösungen von unter 10 ps [Szp+13]. Damit ließen sich mindestens drei zusätzliche Bit pro Puls übertragen. Eine solche Schaltung wird jedoch auch deutlich mehr Platz und Energie benötigen. So geben Szplet u. a. [Szp+13] für ihren Zeitmesser eine Größe von 13,44 mm^2 und einen Energiebedarf von 750 mW auf einem 45 nm-Spartan 6 XC6SLX75 [Spartan6] an. Die Schaltung benötigt etwa 15 % der 74637 Logikelemente des Spartan. Diese Werte sind nicht direkt mit den in dieser Arbeit ermittelten Werten vergleichbar, da das Cyclone II FPGA auf einer wesentlich älteren und nicht so energieeffizienten 90 nm-Technologie basiert. Dennoch lässt sich qualitativ ableiten, dass genauere Zeitmesser in der Regel größer sind und mehr Energie benötigen. Für die PWM-Datenübertragung bedeutet das, dass mit höherem Hardware- und Energieaufwand die Datenrate erhöht werden kann. Der Aufwand steigt allerdings exponentiell, da die Datenrate nur logarithmisch von der Zeitauflösung abhängt.

8.4.3 Bessere Anpassung an die Kanalbandbreite

Der ermittelte Schätzwert für t_{minpuls} zeigt deutlich, dass die PWM-Datenübertragung nicht im optimalen Arbeitspunkt betrieben wird. Aus den Ausführungen in Abschnitt 5.3 in Tabelle 5.1 ergibt sich mit dem Verhältnis $v = t_{\mathrm{minpuls}}/t_{\mathrm{abstand}} = 5$ eine optimale Symbolanzahl $n_{\mathrm{Symbol}} = 8$. Die sich ergebende Symbollänge beträgt

$$T_{\mathrm{Symbol}} = 2 \cdot t_{\mathrm{minpuls}} + t_{\mathrm{abstand}} \cdot (n_{\mathrm{Symbol}} - 1) = 2 \cdot 650\,\mathrm{ps} + 130\,\mathrm{ps} \cdot (8 - 1) = 2{,}21\,\mathrm{ns}.$$

Dafür wäre eine Pulsrate von $1/2{,}21\,\mathrm{ns} \approx 450\,\mathrm{MHz}$ notwendig. Die spektrale Effizienz läge mit $E = 2 \cdot 650\,\mathrm{ps} \cdot 3\,\mathrm{Bit}/2{,}21\,\mathrm{ns} = 1{,}76\,\mathrm{Bit/s/Hz}$ niedriger, als mit einer idealen Bit-seriellen Übertragung möglich wäre. Bei einer minimalen Pulslänge

von 650 ps kann ein, mit $1/650\,\text{ps} \approx 1{,}54\,\text{GHz}$ getakteter Bit-serieller Empfänger eine Datenrate von 1,54 GBit/s erreichen.

Nun könnte der Gedanke aufkommen, dass die PWM-Datenübertragung bei kurzen Kabeln mit hohen Bandbreiten keine Bedeutung hat. Allerdings sei an dieser Stelle darauf hingewiesen, dass das Cyclone II FPGA gar keine Taktfrequenz von 1,54 GHz erreicht. Hier kann die asynchrone Implementierung der Pulsweitenmodulation ihre Vorzüge ausspielen. Daher ist die PWM-Datenübertragung auch auf kurzen Übertragungsstrecken interessant, wenn die Endgeräte in ihrer Taktfrequenz beschränkt sind, sei es aufgrund von schaltungstechnischen Beschränkungen oder aufgrund des Energie-Budgets. Zur Verifikation dieser Aussage, stellt das nächste Kapitel eine Gigabit-Übertragung mit dem Cyclone II FPGA vor.

9 Gigabit-Übertragung mittels Pulsweitenmodulation

In den vorherigen Kapiteln wurde ausführlich der grundsätzliche Aufbau der asynchronen PWM-Datenübertragung beschrieben. Die ersten Ergebnisse versprechen eine Datenrate von 70 MBit/s bei 10 MHz Pulsfrequenz. Aufgrund der hohen Bandbreite des verwendeten Koaxialkabels sollten prinzipiell deutlich höhere Datenraten erreichbar sein. Da in praktischen Anwendungen in der Regel eine zu erreichende Datenrate vorgegeben ist, wir in diesem Kapitel ebenso verfahren. Die zu erreichende Datenrate wird aus verschiedenen Gründen auf 1 GBit/s festgelegt. Einerseits ist das ein runder Wert, der sich gut verkaufen lässt. Andererseits liegt der Wert über der maximalen Bit-seriellen Datenrate des Cyclone II FPGA. Zum Vergleich: Altera spezifiziert für das Senden per LVDS eine maximale Datenrate von 640 MBit/s [CycloneII08, Seite 2-53].

Im Folgenden werden die Maßnahmen beschrieben, die insgesamt zur Erhöhung der Datenrate auf 1 GBit/s geführt haben. Dazu trägt die geschickte Modifikation aller Schaltungsteile bei. Darunter sind die Pulserzeugung und die Symboltabellen. Ein wichtiger Aspekt für die Umsetzung ist, dass die gewählten Parameter bei der Synthese zu einer funktionierenden Schaltung führen. Gerade bei hohen Taktfrequenzen werden die Zeitabhängigkeit zwischen synchronen und asynchronen Schaltungsteilen problematisch. Aufgrund der mangelnden Unterstützung der Synthesewerkzeuge für asynchrone Schaltungen, ist für jede Variation der Parameter eine Synthese und ein praktischer Test erforderlich.

9.1 Architektur

Der grundsätzliche Aufbau der PWM-Datenübertragungsstrecke ist bereits aus Kapitel 6 bekannt. Zum Erreichen der gewünschten Datenrate wird die Wahl der Parameter anhand der Berechnungen aus Abschnitt 5.4 durchgeführt. Die Pulsauflösung der Tapped Delay-Line im Empfänger ist mit etwa 130 ps bekannt. Die Bandbreite der FPGA-Pins liegt laut Datenblatt [CycloneII08, Seite 5-51, Tabelle 5-45] bei etwa 400 MHz. Die Bandbreite wird als „Maximum Clock Toggle Rate“ [CycloneII08, Seite 5-46] bezeichnet. Sie beschreibt die maximale Frequenz eines Rechtecksignals, das übertragen werden kann, und entspricht damit genau der Abschätzung, die auch in dieser Arbeit (z.B. Unterabschnitt 2.2.2) für die

Bandbreite verwendet wird. Aus den Angaben des Datenblattes ergibt sich eine minimale Pulsbreite von 1,25 ns. Für das Verhältnis $v = t_{\text{minpuls}}/t_{\text{abstand}}$ folgt nach Gleichung 6.13

$$v = 1{,}25\,\text{ns}/130\,\text{ps} = 1/2 \cdot (n_{\text{Symbol}} \cdot (\ln n_{\text{Symbol}} - 1) + 1)$$

$$237/13 = n_{\text{Symbol}} \cdot (\ln n_{\text{Symbol}} - 1)$$

Die Lösung für die optimale Symbolanzahl ist $n_{\text{Symbol}} \approx 12$. Aufgrund der Unsicherheiten in der Abschätzung der Bandbreite wird die nächstgrößere Zweierpotenz 16 als Stufenanzahl für die asynchronen Schaltungsteile gewählt. Zunächst wird der Aufbau der Sendeschaltung beschrieben.

9.1.1 PWM-Sender

Aufbau und Funktion

Der bisherige Pulsformer der PWM-Datenübertragung basiert im wesentlichen auf einer einstellbaren, asynchronen Verzögerungsleitung. Er ist in Abbildung 9.1 (a) dargestellt. Ausgehend von der Schaltung mit einer Buffer-Kette ist es möglich, auch statt des unverzögerten Pfades ebenfalls eine konfigurierbare Verzögerungsleitung einzusetzen (siehe Abbildung 9.1 (b)). Der Eingangsvektor besteht nun nicht mehr aus einem n-Bit breiten Wert, sondern aus zwei n-Bit breiten Werten (<zf>,<zs>). Dabei ist <zf> der Vektor für die obere Verzögerungsleitung, die auf die fallende Flanke des Ausgangspulses wirkt, und <zs> ist der Vektor für die untere Verzögerungsleitung, mit Wirkung auf die steigende Flanke. Bei einer alternativen Betrachtungsweise entspricht das einem Eingangsvektor der Pulserzeugung mit der Breite $2 \cdot n$ Bit. Am Ausgang des Logikgatters entsteht ein Puls, dessen Länge der Laufzeitdifferenz der Signale durch die beiden Verzögerungsleitungen entspricht. Aufgrund des gewählten Gatters sind alle Werte nutzbar, bei denen die Verzögerung durch <zf> größer ist, als diejenige durch <zs>. Dann tritt die fallende Signallfanke nach der steigenden Flanke auf, und ein Puls wird

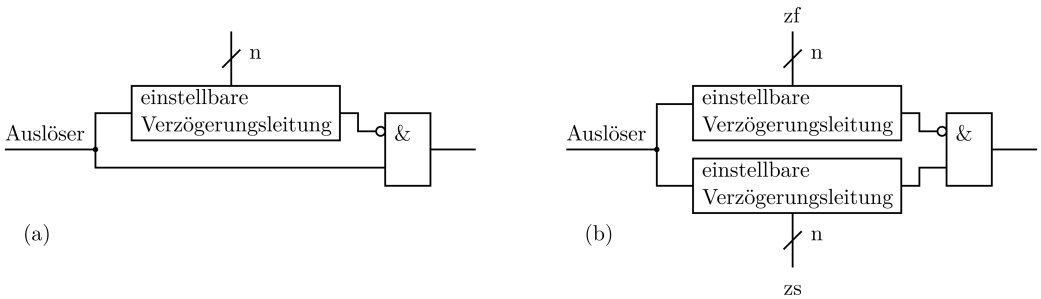

Abbildung 9.1: Die zweite Verzögerungsstrecke ist identisch zur ersten aufgebaut, und quadriert damit die Anzahl der einstellbaren Pulslängen.

erzeugt. Beispielsweise wird die Einstellung (10;0) ungefähr einen Puls der Länge $10 \cdot 130\,\text{ps} = 1300\,\text{ps}$ erzeugen.

Das gemeinsame Verzögern der steigenden und der fallenden Flanke wirkt auf den ersten Blick widersinnig, da eine Eingangsdatenkombination (10;0) den gleichen Ausgangspuls erzeugt, wie die Kombination (11;1). Jedoch sind, aufgrund der Variationen im Herstellungsprozess, die Buffer-Delays nicht exakt identisch. Das bedeutet, dass die Kombinationen (10;0) und (11;1) eben doch unterschiedlich lange Pulse am Ausgang erzeugen. Das bedeutet, dass nicht nur eine Pulsdauermodulation statt findet, sondern bei Nutzung der unteren Verzögerungsleitung gleichzeitig eine parasitäre Pulspositionsmodulation stattfindet. Bei kleinen Verzögerungswerten mag das unproblematisch sein, ist jedoch für das Timing im Empfänger zu beachten.

Jeder Puls, der in den Pulseingang des Senders eingespeist wird, erzeugt am Ausgang des Logikgatters einen Ausgangspuls. Die Anzahl der verschiedenen Pulslängen ist gleich der Anzahl der möglichen Kombinationen der beiden Verzögerungsleitungen. So ergeben sich bei 16 Stufen pro Verzögerungsleitung 256 verschiedene Pulslängen. Unter der Randbedingung, dass <zf> größer oder gleich <zs> sein soll, ergeben sich noch 136 verschiedene Pulslängen.

Zeitauflösung der differentiellen Pulserzeugung

Die differentielle Pulserzeugung ermöglicht zeitliche Auflösungen, die kleiner sind, als die kleinsten Buffer-Delays. Die Erklärung dieser Eigenschaft basiert auf zwei Überlegungen: Einerseits ist der Einstellbereich T_F einer Verzögerungskette etwa

$$T_\text{F} = 16\,\text{Stufen} \cdot 130\,\text{ps/Stufe} \approx 2\,\text{ns}$$

groß. Andererseits können nicht alle Verzögerungen gleich groß sein. Aufgrund der Prozessvariation sind die Verzögerungen aller Verzögerungselemente voneinander verschieden. Daher sind nach den obigen Ausführungen wirklich 136 verschiedene Verzögerungswerte einstellbar. Die mittlere Zeitauflösung t_abstand beträgt

$$t_\text{abstand} \approx 2\,\text{ns}/136 \approx 15\,\text{ps}.$$

Natürlich sind die 136 Verzögerungswerte nicht gleichmäßig verteilt. Allerdings sorgt die Größe der zufälligen Prozessvariationen von deutlich mehr als 15 ps dafür, dass die Intervalle innerhalb des Einstellbereiches ziemlich gleichmäßig verteilt sind. Durch diese feine Einstellungsmöglichkeit kann auf mögliche Sprünge im Verlauf der Empfangswerte, die im vorherigen Kapitel ein Problem waren, reagiert werden. Im Vergleich zur einfachen Verzögerungskette können daher mit dem differentiellen Pulserzeuger mehr Symbole am Empfänger unterschieden werden. Die experimentell ermittelten Werte in Abschnitt 9.3 sprechen ebenfalls für diese Argumentation.

9.1.2 PWM-Empfänger

Für den Empfänger einer Gigabit-Übertragung kommen mehrere Ansätze in Frage. Zuerst einmal ist es sinnvoll, die komplette asynchrone Empfangshardware auch auszunutzen. Das bedeutet, dass die Länge der Tapped Delay-Line an die zu erwartende Pulsrate angepasst wird. Für den Fall der Gigabit-Übertragung entspricht das ungefähr der maximal realisierbaren Taktrate des FPGA. Eine Reduktion der Schaltungsgröße auf einen minimal nötigen Umfang entspannt auch die Zeitanforderungen für das Synthesewerkzeug. Dadurch nimmt die Synthesedauer ab, und es gibt mehr erfolgreiche Synthesedurchläufe. Die in Kapitel 6 vorgestellte Pulslängenmessung per Tapped Delay-Line kann unverändert übernommen werden. Die Implementierung in VHDL mit eingebetteter Timing-Information und der Größen-Definition per VHDL-generic erweist sich hierbei als erstaunlich praktikabel, denn für die Tapped Delay-Line sind keine weiteren Anpassungen erforderlich.

Einsenzähler ohne Addition

Die Aufgabe des Einsenzählers ist, im Ausgabevektor der Tapped Delay-Line Einsen zu zählen. Der in Kapitel 7 beschriebene Aufbau aus Addierern funktioniert für diesen Zweck, lässt aber eine Eigenschaft des Tapped Delay-Line-Vektors außer Betracht: Der Vektor besteht aus einer Reihe von Einsen gefolgt von einer Reihe von Nullen. Daher ist eigentlich keine Addition aller Bits notwendig, sondern es genügt, den Umschlagpunkt zwischen den Nullen und Einsen zu finden. Die Schaltung in Abbildung 9.2 dient genau diesem Zweck.

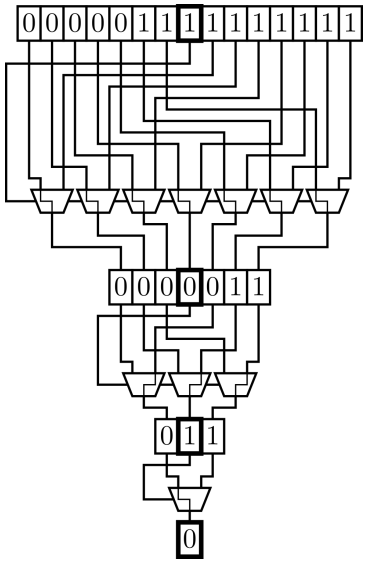

Abbildung 9.2: Schaltungsaufbau zum Einsen zählen mit einem Multiplexerbaum

Auf den ersten Blick sieht die Schaltung aufgrund der vielen Verbindungsleitungen relativ komplex aus. Durch eine rekursive Beschreibung ist die Schaltung jedoch einfach nachzuvollziehen. Der Ergebnisvektor der Tapped Delay-Line wird so angeordnet, dass die Einsen rechts stehen. Ganz unten im Bild, bei der Bit-Breite 1, gibt das Bit direkt die Anzahl der Eins-Bits an. Darüber, bei der Bit-Breite 3, gibt das mittlere Bit an, ob mindestens die beiden rechten Bits Eins sind. Das mittlere Bit entspricht daher in der Binärdarstellung der Einsenanzahl dem 2^1-Bit. Für die höherwertigen Bits wird diese Vorschrift nochmals angewendet.

Umsetzung mit rekursivem VHDL-Quelltext

Für Bit-Breiten der Form $2^x - 1$ kann die Schaltung zur Berechnung der Anzahl Eins-Bit mit folgendem synthetisierbaren, rekursiven VHDL-Code erstellt werden:

```
1  architecture rekursiv
2     function halbieren ( v : std_logic_vector )
3     return std_logic_vector
4     is
5        variable w : std_logic_vector(v'length-1 downto 0) := v;
6        variable ih : integer := w'length/2;
7        variable halb : std_logic;
8     begin
9        -- rekursionsabbruch
10       if w'length = 0 then
11          return w(-1 downto 0);  -- null vector :-)
12       end if;
13       -- entscheide, welche hälfte
14       if w(ih) = '1' then
15          return '1' & halbieren( w(w'length-1 downto ih+1 ) );
16       else
17          return '0' & halbieren( w(ih-1 downto 0) );
18       end if;
19    end halbieren;
20 begin
21    zaehlen: process( e_parallel )
22    begin
23       -- halbieren muss mit einem std_logic_vector
24       -- der länge 2**x-1 aufgerufen werden.
25       anzahl <= halbieren( e_parallel );
26    end process zaehlen;
27 end architecture rekursiv;
```

`e_parallel` ist dabei der Thermometer-Code der Tapped Delay-Line.

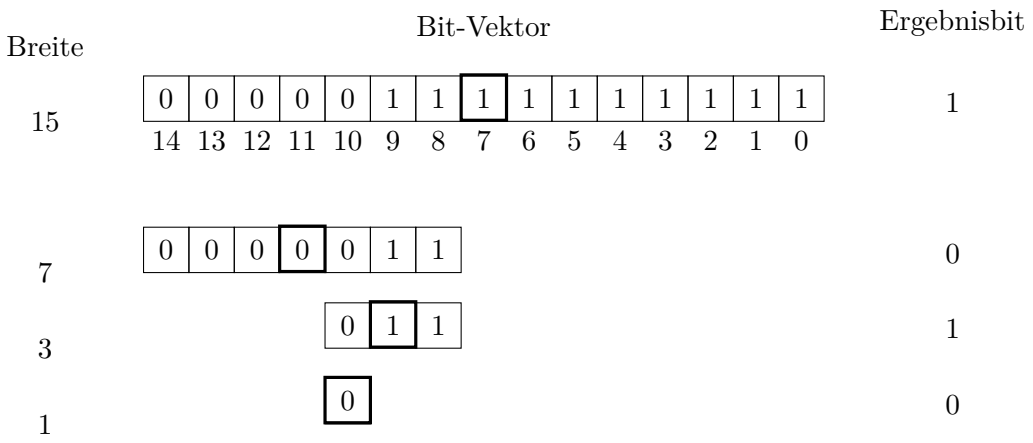

Abbildung 9.3: Einsen zählen mit einem Multiplexerbaum

Beschreibung des Schaltungsverhaltens

Der Ablauf der Auswertung ist in Abbildung 9.3 dargestellt. Als Eingang wird ein Thermometer-Code mit 10 Einsen angenommen. Das mittlere Bit (7) ist 1. Daher sind mindestens 8 Einsen im Bit-Vektor vorhanden und das höchstwertige Ergebnisbit ist ebenfalls 1. Im nächsten Schritt wird der 7 Bit breite Vektor links des Bit (7) betrachtet. Dessen mittleres Bit (11) ist 0. Daraus folgt das nächste Ergebnisbit mit dem Wert 0. Wegen des Bit-Wertes 0 wird im nächsten Schritt der Bit-Vektor rechts des Bit (11) betrachtet. Dessen mittleres Bit (9) ist 1. Im letzten Schritt ergibt sich das niederwertigste Ergebnisbit mit dem Wert 0. Der Ausgangswert ist dann die Bit-Folge 1010, die der dezimalen 10 entspricht.

Einsenzähler als Addiererbaum

Das Zusammenzählen der Einsen ist auch mit einem Addiererbaum möglich. Abbildung 9.4 zeigt einen solchen Baum mit einem Bit-Vektor der 10 Einsen enthält. Der Vorteil des Addiererbaumes gegenüber dem Multiplexerbaum des vorherigen Abschnittes ist, dass die Einsen nicht hintereinander liegen müssen, sondern beliebig über den Bit-Vektor verteilt sein können. Der Einsatz eines Addiererbaumes ist vom Hardware-Aufwand vergleichbar mit der Addiererkette. Die Tiefe des Addiererbaumes steigt logarithmisch mit der Anzahl der Bits. Nach jeder Addiererstufe kann ein Pipeline-Register eingefügt werden. In Kombination mit den auf FPGAs grundsätzlich schnellen Ripple-Carry-Addierern können hohe Taktraten erreicht werden. Selbstverständlich werden durch die Pipeline-Register die Schaltungsgröße und der Energiebedarf vergrößert. Die Anzahl und Position der Pipeline-Register ist demzufolge ein Kompromiss aus Energiebedarf und erreichbarer Taktfrequenz.

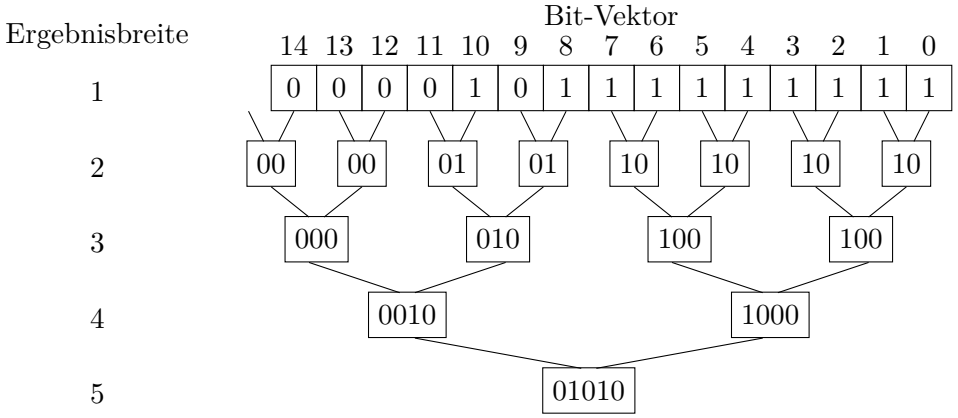

Abbildung 9.4: Schematische Darstellung eines Addiererbaums für 15 Eingangsbits.

Da die erreichbare Taktfrequenz erst im Rahmen der Schaltungssynthese ermittelt wird, erfordert eine optimale Positionierung der Pipeline-Register wiederum praktische Versuche.

9.1.3 Symboltabellen aus parallelen RAM-Blöcken

Die in Kapitel 7 verwendeten RAM-Blöcke sind für Taktfrequenzen bis 250 MHz spezifiziert. Daher müssen die Symboltabellen für Frequenzen über 250 MHz anders implementiert werden. Eine Variante für die Implementierung ist, unter Weiterverwendung der RAM-Blöcke, die Symboltabellen zu parallelisieren. Dabei werden zwei RAM-Blöcke mit identischem Inhalt abwechselnd für die Symbolumsetzung benutzt. Die Pulsrate kann damit prinzipiell von 250 MHz auf 500 MHz verdoppelt werden. Die zusätzliche Logik für die abwechselnde Ansteuerung der RAM-Blöcke verringert jedoch die erreichbare Taktfrequenz.

9.2 Versuchsaufbau der Gigabit-Übertragung

Der Versuchsaufbau ist identisch mit dem in Kapitel 7 vorgestellten Aufbau. Er besteht aus zwei FPGA-Entwicklungsplatinen vom Typ DE2-70, die mit einem 1 m langen Koaxialkabel verbunden sind.

Symboltabelle

Als Symboltabelle dienen die zwei parallelen RAM-Blöcke. Das Blockschaltbild ist in Abbildung 9.5 zu sehen. Die Timing-Analyse gibt für diese Schaltung auf

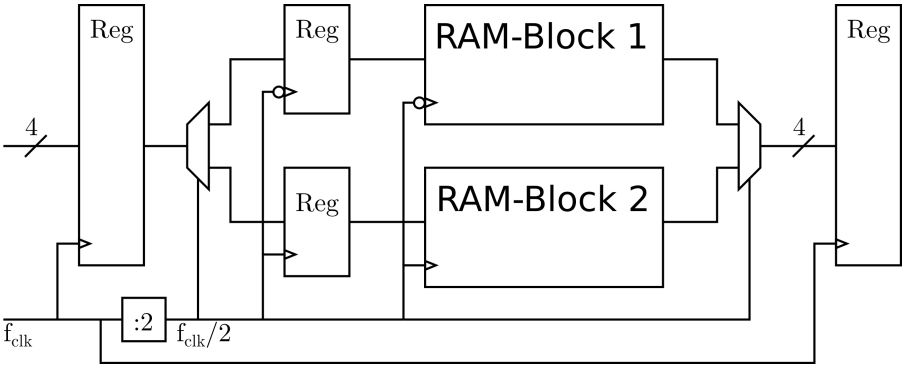

Abbildung 9.5: Symboltabelle als parallele RAM-Blöcke. RAM-Block 1 wird mit der fallenden Flanke des halbierten Taktes ($f_{\text{clk}}/2$) betrieben, und RAM-Block 2 mit der steigenden Flanke.

dem Cyclone II FPGA eine Frequenz f_{clk} von 316 MHz an. Die Pulsrate beträgt also maximal 316 Millionen Pulse pro Sekunde. Damit wäre bei 4 Bit pro Puls eine Datenrate von 1,264 GBit/s erreichbar. Die Pulsperiode beträgt 1/316 MHz ≈ 3,16 ns.

Datenratenabschätzung

Falls nicht alle Symbole genutzt werden können, und nur 3 Bit pro Puls übertragen werden können, würde die Datenrate bei 0,948 GBit/s liegen. Da so die Anforderung von einem Gigabit pro Sekunde verfehlt würde, wird die Pulsrate auf 333 MHz festgelegt. Das bedeutet, dass alle 3 ns ein Puls gesendet wird. Das liegt zwar leicht über dem berechneten Zeitverhalten der Symboltabellen, aber in der Praxis funktioniert die Schaltung trotzdem.

Anzahl der asynchronen Verzögerungsstufen und LVDS-Pins

Für die Pulserzeugung wird die Kombination von zwei 16-stufigen, parallelen Verzögerungsleitungen umgesetzt. Die Pulsmessung erfolgt mit einer Tapped Delay-Line mit ebenfalls 16 Elementen. Für mehr Elemente konnte Quartus trotz einiger Erfolgsmeldungen keine funktionierende Schaltung erzeugen. Für die elektrische Übertragung des Signal werden die selben LVDS-Pin-Paare wie in Kapitel 7 verwendet. Der Sender benutzt mit den Pins `C29` und `C30` das Pin-Paar `LVDS135` als Ausgang. Der Empfänger benutzt die Pins `G26` und `G25`, also das Pin-Paar `LVDS131` als Eingang.

Latenz

Aufgrund der für das Cyclone II FPGA recht hohen Taktfrequenz von 333 MHz sind im Sender und im Empfänger einige Registerstufen für Pipelining notwendig. Die Latenz innerhalb des Senders beträgt 5 Takte bei 333 MHz = 15 ns. Die Latenz im Empfänger beträgt 7 Takte bei 333 MHz = 21 ns. Dazu kommen, wie in Abschnitt 7.2 ausgeführt etwa 5 ns für die Platinen und FPGA-Pins und 5 ns für das Koaxialkabel. In der Summe beträgt die Latenz 51 ns. Das bedeutet, dass die Daten innerhalb von 17 Takten über einen Meter vom Sender zum Empfänger übertragen werden können. Die folgende Tabelle bietet nochmals eine Übersicht der einzelnen Beiträge zur Latenz.

Komponente	Latenz in Takten	Latenz in ns
Sendesymboltabelle	5 Takte	15 ns
Pulsgenerator		2 ns
Sende-FPGA & Platine		5 ns (geschätzt)
Kabel		5 ns
Empfangsplatine & FPGA		5 ns (geschätzt)
Pulslängenmesser	2 Takte	6 ns
Empfangssymboltabelle	5 Takte	15 ns
Summe		51 ns

9.3 Ergebnisse der Gigabit-Schaltung mit 1 m Kabel

Abbildung 9.6 zeigt den Ausgangswert der TDL für ausgewählte Kombinationen von Verzögerungswerten der beiden gegenläufigen Senderverzögerungsleitungen. Die Darstellung der Sendereinstellung erfolgt in der Form (<zf>;<zs>). Dabei entspricht der erste Wert in der Klammer <zf> der ausgewählten Anzahl Verzögerungsstufen der oberen Verzögerungsleitung (siehe Abbildung 9.1). Diese Verzögerung wirkt auf die fallende Flanke des Pulses. Der zweite Wert <zs> gibt die Einstellung der unteren Verzögerungsleitung an, die auf die steigende Flanke des Pulses wirkt.

Die ersten drei Sendepulseinstellungen (0;0), (1;0) und (2;0) führen am Empfänger zu keinem Ergebnis. Der Puls erreicht am Empfänger nicht die Schaltschwelle des Eingangspins. Der nächste Puls (3;0) erreicht den Empfänger, und hat dort eine Länge von sechs Verzögerungsstufen, was einer Pulslänge von etwa $6 \cdot 130\,\text{ps} \approx 0{,}8\,\text{ns}$ entspricht. Die nächsten Pulse (4;0) und (5;0) werden jeweils 2 Tapped Delay-Line-Stufen weiter empfangen. Erst bei den Pulsen (5;0), (6;0) und (7;0) kommt es zu der erwarteten 1:1-Abbildung der Verzögerungsstufen: Jedes zusätzliche Delay im Sender führt zu einem zusätzlichen Delay im Empfänger.

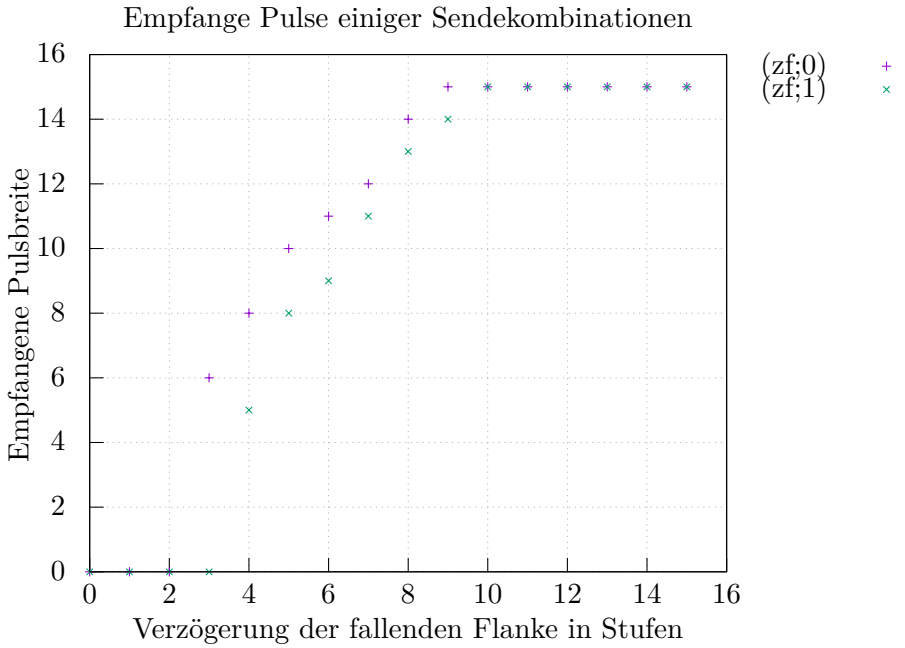

Abbildung 9.6: Die Empfangswerte für die ersten 32 Sendepulse, von (0;0) über (15;0) in violett und (0;1) bis (15;1) in grün.

Der Puls (9;0) löst an der Empfänger-TDL den Wert 15 und damit den Maximalwert aus. Alle längeren Pulse (10;0) bis (15;0) führen zum gleichen Wert. Die Pulse (0;0) bis (2;0) kommen nicht am Empfänger an. Die sieben Pulse (3;0) bis (9;0) erzeugen am Empfänger unterscheidbare Ergebnisse, und sind damit für die Datenübertragung nutzbar.

Die Verkürzung der steigenden Flanke um einen Schritt führt zu einer weiteren Serie von Pulsen des Musters (zf;1). Die 16 Pulse der zweiten Serie sind in Abbildung 9.6 als x in grün dargestellt. Bei ihnen kommt ein Element der unteren Verzögerungsleitung im Sender zum Einsatz. Diese Verzögerung wirkt auf die steigende Flanke des Pulses und daher verkürzend auf die Pulslänge. Bei gleichem zf-Wert, verglichen mit der (zf,0)-Serie in violett, ist jeweils ein kürzerer Puls am Empfänger messbar. Der Sendepuls (4;1) ist der kürzeste Puls, der den Empfänger erreicht. Der Puls ist 5 Verzögerungsstufen lang. Dieser Puls liefert, zusammen mit den Pulsen (6;1) und (8;1) drei zusätzliche Symbole im Empfänger. Die weiteren Symbole leisten keinen Beitrag zur Datenübertragung, da aus der Pulsserie (zf,0) schon Pulse gemessen wurden, die denselben Empfangswert auslösen. Ein Beispiel dafür seien (4;0) und (5;1), die beide einen Empfangswert von 6 auslösen. Die weiteren Puls-Serien wurden vernachlässigt, da kein kürzerer Puls, als der Empfangswert 5 beobachtet werden konnte. Damit ist das Erreichen von 4 Bit je Puls mit dieser Schaltung unmöglich. Für 3 Bit je Puls hingegen liefern

die violetten und grünen Punkte in Abbildung 9.6 bereits genügend verschiedene Empfangswerte. Insgesamt sind am Empfänger die zehn in der folgenden Tabelle dargestellten Symbole unterscheidbar. Daraus werden acht Symbole ausgewählt, die zur Übertragung von drei Bit pro Puls dienen. Die Zuordnung der Datenwerte erfolgt nun entsprechend der Länge der Pulse:

Puls	(4;1)	(3;0)	(4;0)	(6;1)	(5;0)	(6;0)	(7;0)	(8;1)	(8;0)	(9;0)
Länge	5	6	8	9	10	11	12	13	14	15
Daten	-	0	1	2	3	4	5	6	7	-

Für ein Empfangssymbol existiert eine große Anzahl von Delay-Kombinationen, die der Sender verwenden kann. Vorzugsweise nutzt der Sender die Delay-Kombination mit den wenigsten aktiven Elementen im Signalpfad, z.B (4;0) statt (5;1), da mit der Anzahl aktiver Elemente der Einfluss von Temperatur und Spannungsschwankungen steigt.

Die zu erwartenden thermischen Einflüsse können ebenfalls beobachtet werden. So liefert das Sendewort (5;0) direkt nach dem Anschalten den Empfangswert 9. Etwa eine Minute später wird eine 10 empfangen. Abbildung 9.6 ist aus diesem Grund nach 20 Minuten Laufzeit entstanden, nachdem sich die Empfangswerte nicht mehr geändert haben. Nach einem Tag Laufzeit waren die Kalibrierwerte noch dieselben.

Energiebedarf

Im Vergleich zur Schaltung A aus Kapitel 8 hat diese Schaltung nur etwa ein Zehntel der Größe, taktet aber etwa 30 Mal so schnell. Der von Quartus PowerPlay geschätzte Gesamtenergiebedarf beträgt 512,64 mW, davon entfallen 391,52 mW auf die Top-Level-Entity. Die 120 mW der übrigen Schaltungsteile teilen sich wie folgt auf: Die PLL benötigt 45,47 mW. Der Empfänger aus Tapped Delay-Line und Einsenzähler benötigt 10 mW. Der Anteil des Nios II beläuft sich auf 27,16 mW. Die Pulserzeugung benötigt nur 5 mW. Die Einsentabelle des Empfängers benötigt 26,9 mW.

9.4 Bewertung des Gigabit-Versuches

Trotz der vielversprechenden Ergebnisse war kein Dauerbetrieb der Gigabit-Übertragungsstrecke möglich. Das Synthesewerkzeug Quartus hat bei einer ganzen Reihe von Versuchen keine funktionierende Schaltung erzeugen können. Die Ausnahmen von den Timing-Constraints für die asynchronen Schaltungsteile aus Unterabschnitt 6.2.1 waren notwendig, führten aber häufig zu der falschen Aussage eines funktionierenden Schaltkreises. Bei den praktischen Versuchen kam es häufig

vor, dass der Nios II-Prozessor gar nicht anlief, weil ein nicht erkanntes Timing-Problem bestand. Die Aussage des Timing Analyzer war insofern wenig verlässlich. Doch auch der umgekehrte Fall kam vor: Quartus meldete „Timing Requirements not met“ und die Schaltung funktionierte ohne Probleme.

Interessanterweise traten bei Veränderungen der asynchronen Schaltungsteile häufig Fehler in der synchronen Schaltung auf und umgekehrt. Das bedeutet, dass die Entkopplung des Zeitverhaltens durch die Timing-Constraints nicht vollständig gelungen ist. Allerdings ist nach wie vor unklar, welche Entscheidung des Synthesewerkzeuges dieses Problem verursacht. Fakt ist, dass Quartus wenig genug von den asynchronen Schaltungsteilen versteht, um sie durch, eigentlich für synchrone Schaltungen gedachte „Optimierungen“ kaputtzuoptimieren. Das betrifft auch die als unveränderlich angenommenen, explizit spezifizierten LUT-Inhalte, die unter unbekannten Umständen verändert werden.

Ein geplanter Funktionstest mit einem rückgekoppelten Schieberegister konnte in mehrwöchigen Versuchen nicht in eine funktionsfähige Schaltung gebracht werden. Das ist um so trauriger, weil offensichtlich alle einzelnen Schaltungsteile fehlerfrei synthetisiert werden können. Letztendlich bleibt der Synthesevorgang des Fitters etwas Magisches. Die problematische Implementierung der Gigabit-Übertragung stellt darüber hinaus die Grundannahme in Frage, dass die asynchrone PWM-Datenübertragung ohne FPGA-spezifische Platzierungsanweisungen umsetzbar ist. Für das Erreichen der maximalen Datenrate scheint es notwendig zu sein, auf die klassischen, manuell platzierten und gerouteten, asynchronen Schaltungen zurückzugreifen.

10 PWM-Datenübertragung auf langen Kabeln

Die bisherigen Versuche galten der PWM-Datenübertragung auf einem 1 m langen Kabel. Wie in Abschnitt 5.3 ausgeführt, ist die PWM-Datenübertragung jedoch besonders für Kabel mit geringen Bandbreiten geeignet. Diese Eigenschaft trifft insbesondere auf lange Kabel zu. Zur Untersuchung der Eignung beschreibt dieses Kapitel die Verwendung der PWM-Datenübertragung auf Twisted-Pair-Kabeln in Längen von 74,4 m bis 223,2 m. Die Ergebnisse zeigen die Grenzen der Pulserzeugung mit Verzögerungsleitungen hinsichtlich der möglichen Pulslänge. Daher wird in einem ersten Schritt die Länge des Pulserzeugers auf den Maximalwert für das Cyclone II FPGA gesetzt. Darüberhinaus wird ein alternativer Pulserzeuger entwickelt, der die asynchrone Verzögerungsleitung mit einem Zähler kombiniert.

10.1 Bestimmung der Kabeleigenschaften

Das Kabel hat entscheidenden Einfluss auf die Übertragung. Die Art des Kabels und insbesondere seine Länge bestimmen die Auswirkungen auf das Übertragungssignal. Um zu zeigen, dass die PWM bei entsprechender Dimensionierung auch noch bei geringen Bandbreiten Daten übertragen kann, kommen in diesem Versuch möglichst lange Kabel zum Einsatz. Die Wahl des Kabels fiel auf das im Institutsgebäude des Institut für Angewandte Mikroelektronik und Datentechnik in Haus 1 in Warnemünde verlegte Netzwerkkabel. Bei diesem Kabel handelt es sich um verdrillte Doppeladern (englisch Twisted Pair) vermutlich der Kategorie 5 nach TIA/EIA-568 [TIA16]. Die Bestimmung der Kabelparameter erfolgt mit den zwei Testern, einem Fluke DSP-100 CAT 5 Kabeltester und einem Fluke OptiView. Das Fluke DSP-100 wird für die Bestimmung von Kabellänge, Impedanz und Dämpfung verwendet. Leider sorgte ein Defekt des Gerätes dafür, dass nicht alle Kabel vermessen werden konnten. Das Fluke OptiView bestimmt die Kabellänge und die Impedanz. Die Dämpfung wird vom OptiView nicht angegeben. Darüber hinaus dienten ein Signalgenerator und ein Oszilloskop zur Messung einiger Signalspektren.

Tabelle 10.1: Leitungslängen der verwendeten Twisted-Pair-Kabel. Die Reihenfolge entspricht der Verbindung zu 223.2 Metern Gesamtlänge.

Bezeichnung	Typ	Länge
A1	Adapter GPIO - RJ45	1,2 m
H5	Gebäudekabel 1212/5	35 m
P5-6	Patchkabel Serverraum	2 m
H6	Gebäudekabel 1212/6	35 m
PB 1	Patchkabel im Büro	1,4 m
H7	Gebäudekabel 1212/7	35 m
P7-8	Patchkabel Serverraum	3 m
H8	Gebäudekabel 1212/8	35 m
PB 2	Patchkabel im Büro	1,4 m
H9	Gebäudekabel 1212/9	35 m
P9-10	Patchkabel Serverraum	3 m
H10	Gebäudekabel 1212/10	35 m
A2	Adapter GPIO - RJ45	1,2 m
	Summe	223,2 m

10.1.1 Kabellängen

Die genutzten Leitungen der Hausverkabelung verbinden das Büro 1212 mit dem Server-Raum. Die einfache Leitungslänge zwischen beiden Räumen beträgt 35 m. Für die Messungen stehen sechs Leitungen zur Verfügung. Die sechs Leitungen der Hausverkabelung sind nach der Nummerierung der Netzwerkdosen mit H5, H6, bis H10 bezeichnet. Die drei Patch-Kabel P5-6, P7-8 und P9-10 verbinden im Serverraum jeweils die beiden Kabel der Hausverkabelung mit der passenden Nummer. Das bedeutet, P5-6 verbindet H5 mit H6. So ergeben sich drei etwa 74,4 m lange Abschnitte, die vom Büro 1212 aus jeweils einzeln oder miteinander verbunden genutzt werden können. Vom Büro aus war es so möglich, Kabellängen von 74,4 m, 148,8 m und 223,2 m zu nutzen.

Der Anschluss an dass FPGA erfolgt über zwei 1,2 m Leitungsstücke A1 und A2, die an Buchsenleisten angelötet sind. Dabei findet das orange Aderpaar, nach TIA/EIA-568 T568B [TIA16] das Aderpaar 2, Verwendung, welches auf den Kontakten 1 und 2 des RJ45-Steckers liegt. Die maximale gesamte Länge beträgt etwa 223,2 m aus 13 Leitungsstücken. Eine Übersicht aller verwendeten Kabel mit Angabe ihrer Länge ist in Tabelle 10.1 zu finden.

10.1.2 Impedanz und Dämpfung

Die Dämpfung wird bei einer Frequenz von 100 MHz bestimmt. Die Messergebnisse sind Tabelle 10.2 dargestellt. Ein Defekt am Fluke DSP-100 sorgte dafür,

Tabelle 10.2: Die gemessene Dämpfung ausgewählter Leitungskombinationen.

Leitungen	Länge	Impedanz	Dämpfung
PB 1	1,4 m	100 Ω	0,4 dB
PB 2	1,4 m	100 Ω	0,4 dB
PB 1+H5+P5-6+H6+PB 2	74,9 m	106 Ω	13,6 dB

dass keine weiteren Impedanzmessungen möglich waren. Glücklicherweise gibt das letzte Messergebnis die Eigenschaften einer Kabelschleife vom Büro zum Server-Raum und zurück zum Büro wieder. So kann auf die anderen Kabel hochgerechnet werden. Es ist zu erwarten, dass sich die anderen beiden Schleifen ähnlich verhalten, da die gleichen Kabel zum Einsatz kommen. Die geschätzte Dämpfung bei 148,8 m beträgt etwa $2 \cdot 13{,}6\,\text{dB} = 27{,}2\,\text{dB}$. Die Dämpfung des 223,2 m langen Kabels liegt dementsprechend bei $3 \cdot 13{,}6\,\text{dB} = 40{,}8\,\text{dB}$. Die Dämpfung von $40{,}8\,\text{dB} = 20\log_{10}(a)$ entspricht einem Faktor a der Signalamplitude von

$$a = 10^{-\frac{40.8}{20}} \approx 0{,}01$$

Wird also ein Sinussignal mit einer Frequenz von 100 MHz und einer maximalen Amplitude von 1 V an einem Kabelende eingespeist, ist nach 223,2 m, am anderen Kabelende eine Amplitude von 10 mV messbar.

10.1.3 Signalform am 223,2 m Kabel

Zur Bestätigung dieser Abschätzung wurden Messungen mit einem Signalgenerator und einem Oszilloskop auf der 223,2 m-Strecke durchgeführt. Als Eingangsignal dient eine nullsymmetrische Sinusspannung mit einer Amplitude von 1 V. Am anderen Kabelende wurde mit dem Oszilloskop die Signalamplitude bestimmt. Auffallend an Abbildung 10.1 ist dabei, dass schon bei ausgeschaltetem Signalgenerator eine Sinusschwingung von etwa 160 mV Spitze-Spitze-Amplitude mit einer Freuquenz von etwa 130 kHz erkennbar ist. Bei Überlagerung des Störsignals mit dem 100 MHz-Sinus hat dieser noch eine Amplitude von 16 mV. Die Dämpfung beträgt demzufolge $20\log_{10}(1/0{,}016) \approx 36\,\text{dB}$. Sie ist also etwas geringer, als die Schätzung im vorigen Abschnitt. Trotz der relativ starken Störungen ist, wie nachfolgend beschrieben, die PWM-Datenübertragung erfolgreich verlaufen.

Die eigentlich interessante Größe ist die minimal mögliche Pulsbreite t_{minpuls}, die für Übertragung nutzbar ist. Das ist nachfolgend insbesondere für die Parametrierung des Versuchsaufbaus von Bedeutung. Im Vorgriff auf die Ergebnisse in Abschnitt 10.4 ist in Abbildung 10.2 der Signalverlauf dargestellt, der sich ergibt, wenn eine 6,1 MHz-Rechteckschwingung mit einer Amplitude von 400 mV eingespeist wird. Das Bild zeigt, dass die Signalamplitude von 80 mV etwa genauso groß ist, wie diejenige des 130 kHz-Rauschens bzw. -Störsignals. Daraus folgt, dass ein Augendiagramm dieses Signals vollständig geschlossen wäre. Im Vergleich zu

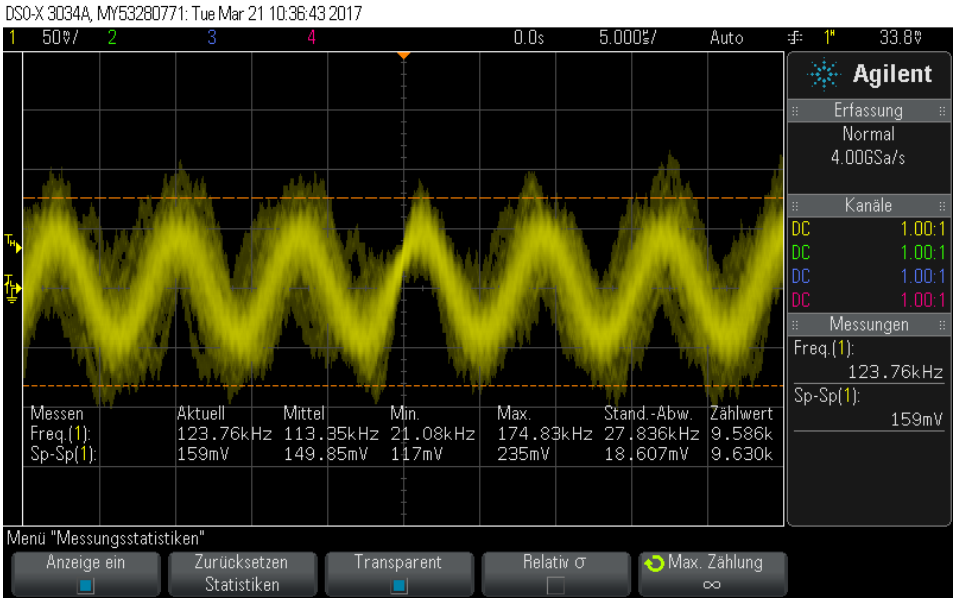

Abbildung 10.1: Rauschen am Ende des 223,2 m Kabels bei ausgeschaltetem Signalgenerator.

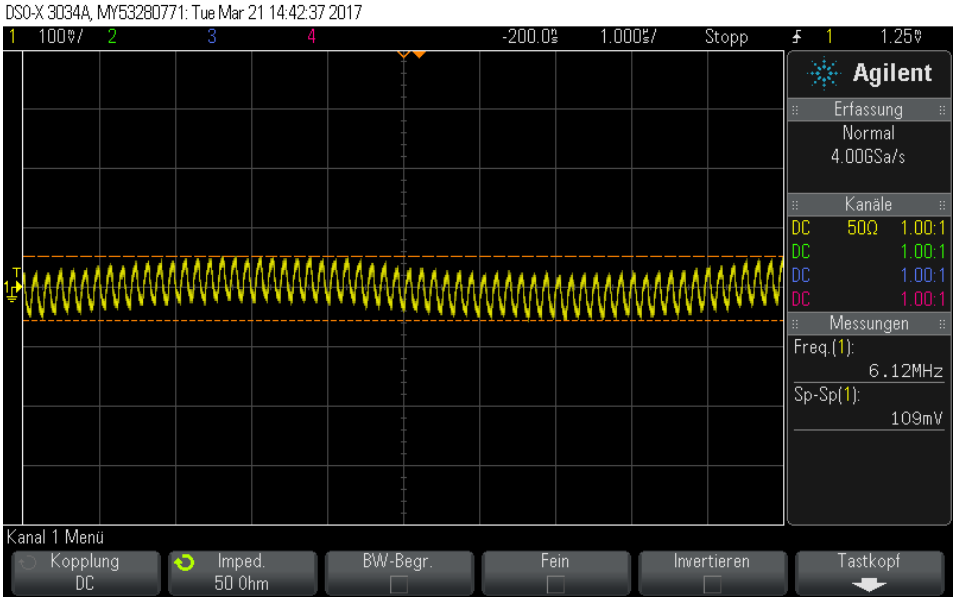

Abbildung 10.2: Verlauf des Empfangssignals. Erkennbar ist das eigentliche 6,1 MHz-Signal mit etwa 80 mV Amplitude aufmoduliert auf das 130 kHz-Störsignal.

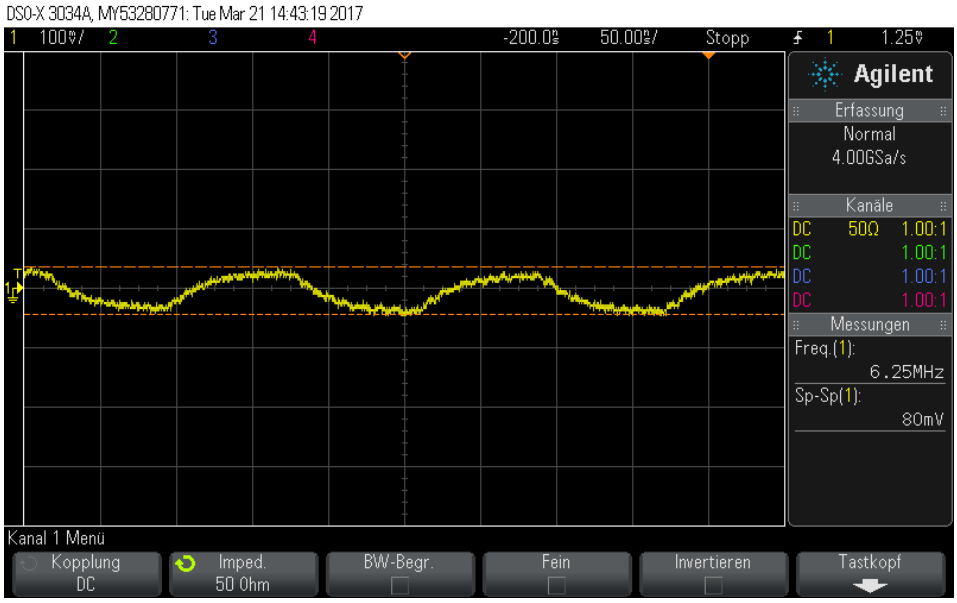

Abbildung 10.3: Ausgehend vom 6,1 MHz-Rechtecksignal mit 400 mV als Eingangssignal, sind nach 223,2 m noch 80 mV Amplitude erkennbar.

Abbildung 10.1 ist die Amplitude halbiert, da der 50 Ω-Abschlusswiderstand am Oszilloskop aktiviert ist. Die genaue Signalform des 6,1 MHz-Signals ist als Ausschnitt von Abbildung 10.2 in Abbildung 10.3 dargestellt. Die 130 kHz-Schwingung ist in dem Ausschnitt nicht erkennbar, da der Ausschnitt nur 500 ns breit ist. Der Signalverlauf der Pulse erinnert an eine (e^{-t})-Funktion. Gut erkennbar ist auch das überlagerte Rauschen, mit etwa 10 mV Amplitude, und wie es sich als Jitter auf die Frequenzmessung des Oszilloskops auswirkt (rechts im Bild: 6,25 MHz).

10.2 Modifikationen der Modulatorarchitektur

In Kapitel 5 ist ausgeführt, dass die PWM-Datenübertragung vielfältig einsetzbar ist, und sich an verschiedene unbekannte Kabel anpassen kann. Dies wird mit Hilfe der in Kapitel 6 vorgestellten Architektur auf einem 74,4 m Kabel gezeigt. Darüberhinaus werden zwei Ideen für Schaltungsanpassungen an die niedrige Bandbreiten erläutert. Eine dieser Ideen, nämlich der hybride Pulsgenerator wird im FPGA implementiert, und in seinen Eigenschaften mit dem bisherigen Pulsgenerator verglichen.

10.2.1 Pulsgenerator

Niedrige Bandbreiten erfordern lange Pulse. Dadurch wird das Modulationsfenster für große Symbollängen sehr lang. Da der Pulsgenerator das gesamte Modulationsfenster abdecken muss benötigt er einen großen Einstellbereich. Die bisher vorgestellten Pulsgeneratoren bestehen vollständig aus asynchronen Elementen. Das EP2C70 FPGA hat 50 LAB-Zeilen mit 16 LUT je LAB. Die längste Carry-Chain erstreckt sich also über 800 Logikelemente. Das erste Logikelement enthält kein Carry-In, und das letzte kein Carry-Out. Daraus ergibt sich das Maximum von 798 Elementen in der Verzögerungskette. Diese ergeben einen Einstellbereich von etwa $130\,\text{ps} \cdot 798 \approx 104\,\text{ns}$. Das Zeitverhalten des dann notwendigen großen Eingangsmultiplexers (siehe Abschnitt 6.1) kann zusätzliche Pipeline-Register erfordern. Alternativ kann ein hybrider Pulsgenerator mit einer wesentlich kürzeren Verzögerungskette zum Einsatz kommen.

Hybrider Pulsgenerator

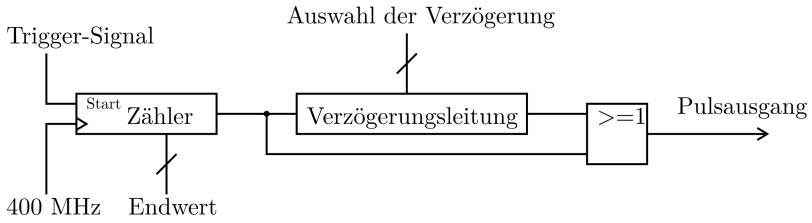

Abbildung 10.4: Der hybride Pulsgenerator.

Der hybride Pulsgenerator ist in Abbildung 10.4 dargestellt. Er besteht aus drei wesentlichen Teilen: der Eingangspulserzeugung, der Verzögerungsleitung und der Signalverknüpfung. Der hybride Pulsgenerator unterscheidet sich vom asynchronen Pulsgenerator dadurch, dass der Trigger-Puls einen Zähler steuert, der einen mehrere Takte langen Puls in die Verzögerungsleitung einspeist. Dies ermöglicht beliebig lange Pulse. Die Eingangspulserzeugung ist eine synchrone Schaltung, die mittels eines Zählers einen Puls erzeugt, der eine ganze Anzahl Takte lang ist. Der Endwert stellt dabei die Pulslänge in Anzahl Takten ein. Die asynchrone Verzögerungsleitung wurde bereits ausführlich in Abschnitt 6.1 beschrieben, und wird hier unverändert übernommen. Die Signalverknüpfung findet durch ein ODER-Gatter statt. Aufgrund der Logikfunktion des ODER-Gatters erfolgt eine Pulsverlängerung um die eingestellte Verzögerung.

10.2.2 Pulsempfänger

Für den Pulsempfänger gibt es zwei naheliegende Varianten. Variante Eins ist ein ebenfalls hybrider Empfänger, der als Tapped Delay-Line mit festem Takt arbeitet. Die Vor- und Nachteile eines hybriden Pulsempfängers werdem im folgenden Absatz erläutert. Variante Zwei ist die schon aus Kapitel 7 bekannte Tapped Delay-Line. Deren Parameter werden an die großen Pulslängen angepasst.

Hybrider Pulsempfänger

Ein hybrider Empfänger bietet, vergleichbar mit dem hybriden Pulserzeuger, Vorteile beim Platzbedarf der Schaltung, die sich letztlich auch positiv auf den Energiebedarf auswirken. Anders als der Sender muss jedoch die Empfängerschaltung den ankommenden Puls einsynchronisieren. Der dafür notwendige Aufwand kann den Platzbedarf der eigentlichen Schaltung um eine Vielfaches überschreiten. Wu und Shi [WS08] haben das bei der Vorstellung des Wave–Union–TDC gezeigt. Darüberhinaus zeigten sie, dass eine relativ komplexe Kalibrierschaltung notwendig ist. Die Ursache liegt im Verlust des Bezuges der Pulslänge zu einem geometrisch eindeutigen Ort im FPGA, im Gegensatz zur klassischen Tapped Delay-Line. Die steigende und fallende Flanke können in einem beliebigen Abstand zum Zählertakt auftreten. Das führt insbesondere dazu, dass die Empfangsschaltung die Zeiten der Pulslänge bestimmen müsste. Daher wäre eine zweistufige Kalibrierung notwendig, die zuerst den Empfänger mit statistisch bekannten Eingangsdaten kalibriert, und im Anschluss die Sendeschaltung mit dem Empfänger vermisst. Neben dem komplexen Ablauf dieser Kalibrierung, übersteigt der Hardware-Aufwand der Kalibrierlogik die eigentliche Zeitmessschaltung bei weitem. Darüberhinaus wird die gesamte Schaltung ständig mit 400 MHz betrieben, selbst wenn kein Eingangssignal anliegt. Der daraus resultierende Platz- und Energiebedarf muss bei einer Implementierung berücksichtigt werden.

Die maximale Länge der Tapped Delay-Line

Die zweite Variante für den Pulsempfänger ist die lange Tapped Delay-Line. Im Cyclone II FPGA lassen sich Tapped Delay-Lines mit bis zu 800 Elementen im Carry-Pfad realisieren. Diese 800 Elemente ermöglichen im Idealfall eine Unterscheidung von 800 Symbolen. Daher kann ein solcher Empfänger maximal

$$(\log_2 800)\text{Bit/Symbol} \approx 9{,}64\,\text{Bit/Symbol}$$

übertragen. Mit der bekannten Zeitauflösung t_{abstand} von etwa 130 ps, errechnet sich daraus anhand der Formel für die datenratenoptimale Bit-Anzahl

$$\frac{t_{\text{minpuls}}}{t_{\text{abstand}}} = \frac{1}{2}\left(1 + 2^b(-1 + b \cdot (\ln 2))\right) \tag{10.1}$$

durch Umstellen nach t_{minpuls}, die minimale Pulsbreite, bis zu der die Schaltung die optimale Datenrate erreichen kann.

$$t_{\text{minpuls}} = \frac{t_{\text{abstand}}}{2}\left(1 + 2^b\left(-1 + b \cdot (\ln 2)\right)\right)$$

$$t_{\text{minpuls}} = \frac{t_{\text{abstand}}}{2}\left(1 + n_{\text{Symbol}}(-1 + \ln n_{\text{Symbol}})\right)$$

$$t_{\text{minpuls}} = \frac{130\,\text{ps}}{2}\left(1 + 800(-1 + \ln 800)\right)$$

$$t_{\text{minpuls}} \approx 65\,\text{ps}\cdot(1 + 800(-1 + 6.685)) = 65\cdot(1 + 800 \cdot 5.685)\,\text{ps} = 65\cdot(1 + 4548)\,\text{ps}$$

$$t_{\text{minpuls}} \approx 65 \cdot 4549\,\text{ps} \approx 295{,}7\,\text{ns}$$

Die Schaltung ist also bis zu minimalen Pulsbreiten von etwa 300 ns einsetzbar. Das entspricht einer Bandbreite von $\frac{1}{2\cdot 300\,\text{ns}} \approx 1{,}7\,\text{MHz}$.

10.3 Versuchsaufbau

Die Messungen wurden jeweils für die Kabellängen 74,4 m, 148,8 m und 223,2 m durchgeführt. Die Leitungsstücke A1 und A2, die mit den GPIO Pins der Entwicklungsplatine DE2-70 verbunden werden, sind direkt mit Buchsenleisten auf kleinen Lochrasterplatinen aufgelötet. Im Rahmen der folgenden Messungen kommen zwei verschiedene Schaltungsvarianten zum Einsatz. Variante A ist identisch mit dem Aufbau aus Kapitel 7. Variante B beinhaltet im Sender den neu vorgeschlagenen hybriden Pulserzeuger und im Empfänger eine Tapped Delay-Line mit 800 Elementen.

10.3.1 Schaltung A

Der Pulsgenerator besteht aus einer Carry-Chain mit 256 Verzögerungselementen. Der Pulsempfänger basiert auf einer 256-stufigen Tapped Delay-Line. Zu Vergleichszwecken wurden mit dem asynchronen Pulserzeuger mit 256-Elementen Messungen über 1 m und 74,4 m durchgeführt.

10.3.2 Schaltung B

Als Pulserzeuger dient ein hybrider Sender aus 400 MHz-Zähler und 32-stufiger einfacher Verzögerungsstrecke. Die 32 Verzögerungselemente erreichen eine Verzögerung von etwa $32 \cdot 130\,\text{ps} = 4{,}16\,\text{ns}$, und decken eine Periode des 400 MHz-Signals sicher ab. Der kürzeste erzeugbare Puls (ein Takt ohne Verzögerung) ist 2,5 ns lang. Die Trigger-Frequenz ist einstellbar bis maximal 50 MHz, was für die abgeschätzte Bandbreite sicher ausreicht. Als Pulsempfänger kommt eine Tapped Delay-Line mit 800 Verzögerungselementen zum Einsatz.

10.3.3 Latenz

Die Latenz der Datenübertragung wird durch das Kabel dominiert. Bei einer Länge von 223,2 m beträgt die Signallaufzeit $\frac{223{,}2\,\text{m}}{2 \cdot 10^8\,\text{m}\,\text{s}^{-1}} = 115 \cdot 10^{-8}\,\text{s} = 1{,}15\,\mu\text{s}$. Vergleichsweise schnell erfolgt die Erzeugung des Pulses im Sender mit 200 ns bei 5 MHz Pulsrate. Der Empfänger benötigt nach dem Empfang des Pulses noch einen weiteren Pulstakt, also 200 ns für die Verarbeitung. Dann kann der Nios II-Prozessor die Daten mit seinem Systemtakt von 50 MHz auslesen. Die Gesamtverzögerung liegt demzufolge bei 1,5 µs.

10.4 Ergebnisse

Im Folgenden werden die Kalibrierungsergebnisse der PWM-Datenübertragung für verschiedene Kabellängen vorgestellt. Zu Beginn steht eine Messung mit der aus Kapitel 7 bekannten Schaltungsvariante A.

10.4.1 256-stufige PWM mit 74,4 m Kabel

Abbildung 10.5 stellt die Empfangswerte der Schaltungsvariante A dar. Die grüne Kurve zeigt die Empfangswerte nach dem 74,4 m-Kabel. In Violett ist das schon aus Kapitel 8 bekannte Ergebnis über 1 m Koaxialkabel zum Vergleich dargestellt. Bei einer Testmessung mit der doppelten Kabellänge von 148,8 m kam kein Puls am Empfänger an.

Der 256-stufige Sender erreicht nach 74,4 m den Empfänger mit den Pulsen 122 bis 255. Der längste, am Empfänger gemessene Puls ist 154 Elemente lang. Die Anzahl der eindeutig erkennbaren Pulse liegt bei 110. Damit sind $\log_2 110 = 6{,}78 > 6$ Bit pro Symbol übertragbar. Die Datenrate beträgt $6\,\text{Bit/Symbol} \cdot 30 \cdot 10^6\,\text{Symbole/s} = 180\,\text{MBit/s}$.

Der kürzeste Puls, der den Empfänger erreicht hat eine Dauer von 12,5 ns. Zum Vergleich: Mit einer Bit-seriellen Übertragung ließen sich damit $1\,\text{Bit}/(12{,}5\,\text{ns}) =$

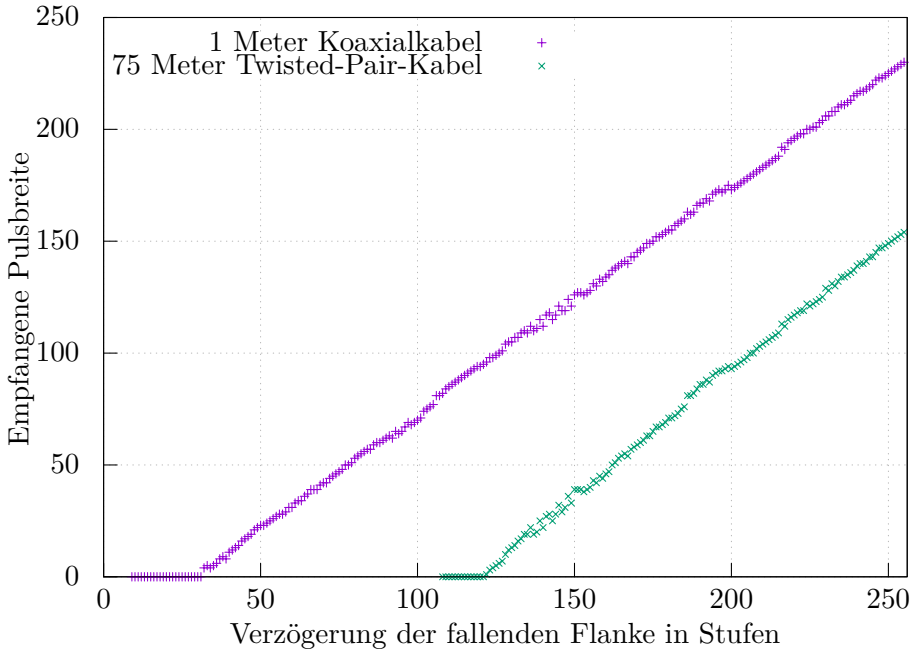

Abbildung 10.5: Messergebnisse der asynchronen Sendervariante bei einer Übertragung über 74,4 m. Zum Vergleich sind die Werte für das 1 m-Kabel eingezeichnet.

80 MBit/s übertragen. Das wiederum entspricht einer Bandbreiteanforderung von mindestens $B_{74{,}4\,\mathrm{m}} = 1/(2 \cdot 12{,}5\,\mathrm{ns}) = 40\,\mathrm{MHz}$, wenn ideales Tiefpassverhalten angesetzt wird.

10.4.2 Ergebnisse des hybriden Pulsgenerators

Abbildung 10.6 zeigt die Empfangswerte der Schaltungsvariante B für die drei verschieden Kabellängen 74,4 m, 148,8 m und 223,2 m. Der hybride Pulsgenerator erzeugt Pulse bis zu einer Dauer von 162,5 ns. Damit kann der Sender den Messbereich der Empfänger-Tapped Delay-Line bei Entfernungen bis 223,2 m ausnutzen.

Die violette Kurve in Abbildung 10.6 zeigt die Messergebnisse über 74,4 m. Der kürzeste Puls, der der den Empfänger erreicht, ist mit 12 ns Länge der Puls mit dem Eingangwert 120. Die Pulslänge liegt sehr dicht bei den 12,5 ns, die für die Schaltungsvariante A gemessen wurden. Am Empfänger ist dieser Puls noch 25 Verzögerungselemente oder 3,25 ns lang. Der Puls mit dem Eingangwert 1162 erreicht den maximalen Empfangswert von 799 bei einer Pulslänge von 92,8 ns. Die grüne Kurve stellt die Kalibrierdaten über 148,8 m dar, die blaue Kurve zeigt die Werte für das 223,2 m-Kabel. Die mit dem Oszilloskop bestimmten Pulslängen

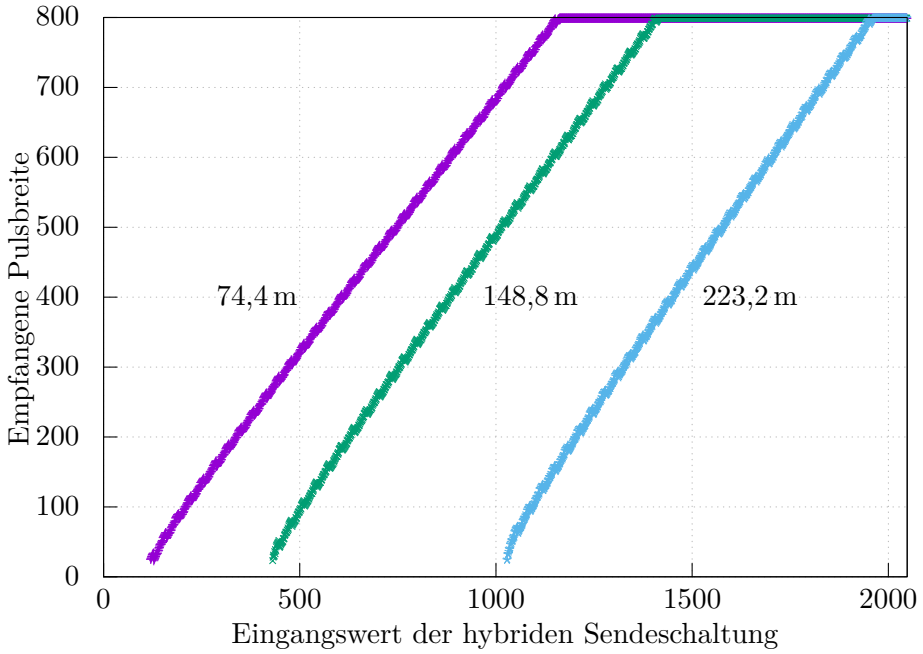

Abbildung 10.6: Empfangswerte der Schaltung B mit den Kabellängen 74,4 m (violett), 148,8 m (grün) und 223,2 m (blau).

des hybriden Pulsgenerators sind in Tabelle 10.3 jeweils für Anfang und Ende aller drei Kurven aufgeführt. Auffällig ist dabei, dass kein Puls empfangen wurde, der im Empfänger das Ergebnis 0 auslöst. Eine mögliche Ursache, liegt in der Leitungsführung innerhalb des Empfänger-FPGA. Diese kann gegenüber der vorher untersuchten Schaltungsvariante A durchaus Abweichungen im einstelligen Nanosekundenbereich aufweisen. Ein gewisser Versatz ist auch zwischen verschiedenen Synthesedurchläufen festzustellen. Die Problematik der deterministischen Schaltungserzeugung wird in Kapitel 11 ausführlich behandelt.

Tabelle 10.3: Am Sender gemessene Pulslängen.

Eingangswert	Pulslänge	Beschreibung
120	11,8 ns	kürzester Puls für 74,4 m
431	35,8 ns	kürzester Puls für 148,8 m
1019	82,3 ns	kürzester Puls für 223,2 m
1162	92,8 ns	Puls, der am Empfänger 799 erreicht bei 74,4 m
1405	112,3 ns	Puls, der am Empfänger 799 erreicht bei 148,8 m
1962	155,3 ns	Puls, der am Empfänger 799 erreicht bei 223,2 m
2047	162,5 ns	längster Puls, des hybriden Senders

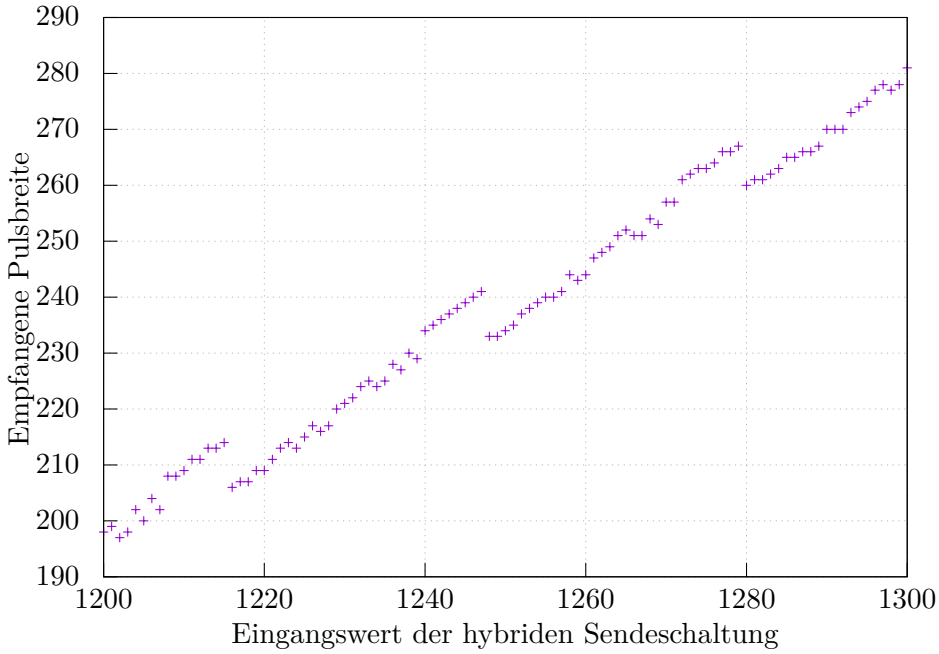

Abbildung 10.7: Ausschnitt aus der Empfangskurve der 223,2 m-Messung.

Senderauflösung

Zur Darstellung der Auflösung des hybriden Senders wird in Abbildung 10.7 ein Ausschnitt aus Abbildung 10.6 betrachtet. Gut erkennbar sind jeweils 32 zusammenhängende Schritte mit monoton steigenden Werten, die von Sprüngen nach unten unterbrochen werden. Die 32 zusammenhängenden Werte werden durch die Verzögerungskette mit ihren 32 Elementen erzeugt. Die Sprünge nach unten entstehen durch das Weiterzählen des Zählers. Anhand des Ausschnitts in Abbildung 10.7 ist gut zu sehen, dass bei weitem nicht alle Empfängerpulse genutzt werden. Je nach Kabellänge kann der Empfänger zwischen 550 und 590 Symbole unterscheiden.

Datenrate

Tabelle 10.4 schätzt anhand der Messergebnisse die maximale Datenrate ab. Dies geschieht für jede der drei Kabellängen in je einer Spalte. Anhand der Werte für 223,2 m werden nachfolgend die Ergebnisse beschrieben. Anhand von Abbildung 10.6 lässt sich der kürzeste empfangbare Puls identifizieren. Es ist der Puls mit dem Eingangswert 1019. Mit dem Oszilloskop wurde seine Länge am Senderausgang zu 82,3 ns bestimmt. Mit Hilfe dieser minimalen Pulslänge t_{minpuls}, lässt sich die Bandbreite B abschätzen $B = 1/(2 \cdot 82{,}3\,\mathrm{ns}) = 6{,}1\,\mathrm{MHz}$. Ebenso kann die

Tabelle 10.4: Abschätzung der erreichbaren Datenrate

	74,4 m	148,8 m	223,2 m
kürzester Puls t_{minpuls}	11,8 ns	35,8 ns	82,3 ns
Bandbreite[1]	42,4 MHz	14 MHz	6,1 MHz
Bit-serielle Datenrate	84,7 MBit/s	28 MBit/s	12,2 MBit/s
unterscheidbare Symbole	588	582	552
Pulsstellbereich	81 ns	76,5 ns	73 ns
mittleres t_{abstand}	145 ps	131 ps	132 ps
$v = t_{\text{minpuls}}/t_{\text{abstand}}$	90,8	275	633
optimale Symbolanzahl	59	140	274
spektrale Effizienz	4,44 Bit/s/Hz	5,67 Bit/s/Hz	6,66 Bit/s/Hz
theoretische Datenrate	188 MBit/s	79 MBit/s	40 MBit/s
b_{ganz}	6	7	8
n_{ganz}	64	128	256
Sendesymbol(n_{ganz})	221	625	1431
Empfangspuls(n_{ganz})	114	199	385
Sendepulslänge(n_{ganz})	19,8 ns	50,8 ns	113,8 ns
T_{Symbol} (n_{ganz})	31,6 ns	86,8 ns	196,1 ns
Pulsrate	$31{,}6 \cdot 10^6$ Pulse/s	$11{,}5 \cdot 10^6$ Pulse/s	$5{,}1 \cdot 10^6$ Pulse/s
R (n_{ganz})	188 MBit/s	81 MBit/s	41 MBit/s

maximale Bit-serielle Datenrate angegeben werden $1Bit/82{,}3\,\text{ns} = 12{,}2\,\text{MBit/s}$. Die Anzahl eindeutig unterscheidbarer Symbole für die Schaltung B liegt bei 552. Diese 552 Symbole verteilen sich auf einen Pulslängenbereich von 82,3 ns bis 155,3 ns. Da dieser $155{,}3\,\text{ns} - 82{,}3\,\text{ns} = 73\,\text{ns}$ breite Bereich 552 Symbole enthält, ist der mittlere Symbolabstand $t_{\text{abstand}} = 73\,\text{ns}/552 = 132\,\text{ps}$, was ziemlich genau der Herstellerangabe für die Verzögerung im Carry-Pfad von 129 ps entspricht. Entsprechend der theoretischen Betrachtung in Abschnitt 5.3 lässt sich unter Annahme einer Zeitauflösung $t_{\text{abstand}} = 130\,\text{ps}$ das Verhältnis $v = t_{\text{minpuls}}/t_{\text{abstand}} = 633$, und daraus die optimale Symbolanzahl $n_{\text{opt}} = 274$ und die maximale spektrale Effizienz $E = 6{,}67\,\text{Bit/s/Hz}$ bestimmen. Aus dem Produkt von Bandbreite und spektraler Effizienz folgt die theoretisch mögliche Datenrate $R(n_{\text{opt}}) = 40\,\text{MBit/s}$.

Die optimale Bit-Anzahl pro Symbol liegt bei $\log_2(274) \approx 8{,}1$. Da die vorgestellte Hardware auf ganze Bit pro Symbol beschränkt ist, wird der gerundete Wert b_{ganz} von 8 Bit pro Symbol verwendet. Dafür muss der Empfänger $n_{\text{ganz}} = 256$ Symbole unterscheiden. Das nach der Kalibrierung bestimmte, 256. Symbol ist das Sendesymbol 1431. Dies erzeugt den Empfangswert 385. Dieses Symbol hat am Sender eine Pulslänge von 113,8 ns. Addiert man diese zur minimalen Pulslänge, so ergibt sich die Symboldauer von 196,1 ns. Als Reziprokes der Symboldauer lässt sich die Pulsrate angeben, die letztlich die Frequenz des Trigger-Pulses zu 5,1 MHz bestimmt. Das Produkt aus Pulsrate und Bit-Anzahl pro Symbol ergibt dann die Datenrate von 40,8 MBit/s. Zur Veranschaulichung zeigt Abbildung 10.8

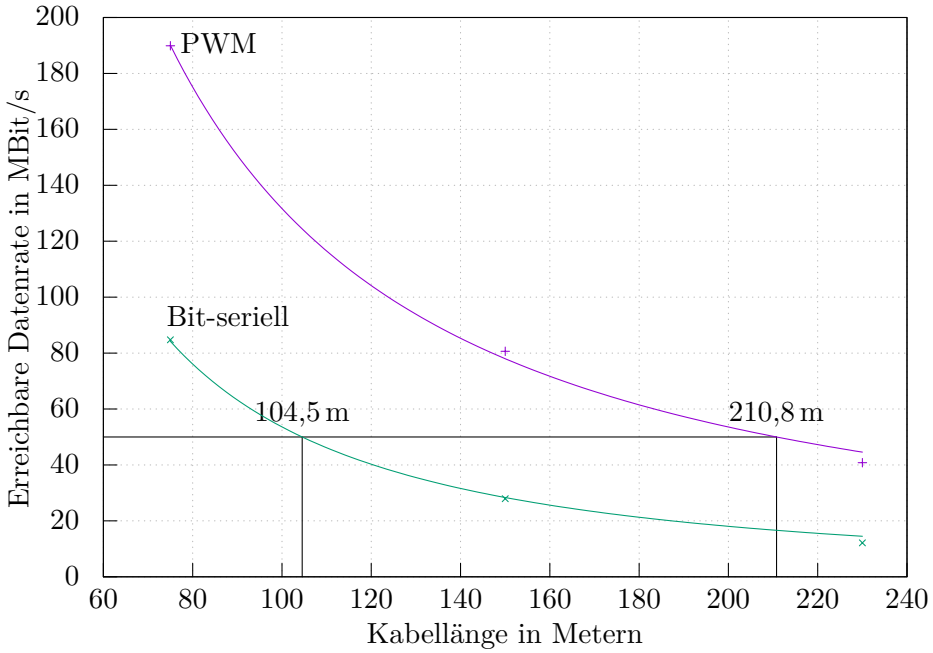

Abbildung 10.8: Datenrate der PWM-Datenübertragung (violett) im Vergleich zur Bit-seriellen Übertragung (grün).

die Datenrate der Pulsweitenmodulation und die Bit-serielle Datenrate über der Entfernung. Daran ist zu erkennen, dass die PWM-Datenübertragung bei gleicher Datenrate von z.B. 50 MBit/s die doppelte Entfernung zurücklegen kann, wie die Bit-serielle Übertragung, nämlich 210,8 m im Vergleich zu 104,5 m.

Interessanterweise liegt die berechnete Datenrate R (n_{ganz}) geringfügig über der theoretischen Datenrate. Die wahrscheinlichste Ursache liegt in Messfehlern bei den Größen, die zur Bestimmung der theoretischen Datenrate verwendet wurden.

Leistungsbedarf

Tabelle 10.5 stellt den Leistungsbedarf der drei in dieser Arbeit verwendeten Schaltungsvarianten gegenüber. Schaltung A ist die in Kapitel 7 vorgestellte Schaltung. Schaltung B ist die in diesem Kapitel entwickelte Schaltung mit dem hybriden Pulsformer für lange Kabel. Zuletzt ist Schaltung C die in Kapitel 9 verwendete kurze Variante der Verzögerungsketten mit 16 Elementen und einer Pulsrate von 333 MHz. Der Energiebedarf der Schaltungsvariante B ist deutlich höher, als derjenige der Schaltung A. Die Tapped Delay-Line hat aufgrund ihrer Länge von 800 Elementen einen Energiebedarf von 80 mW, d.h 0,1 mW pro Verzögerungselement. Die PLL benötigt mit 63 mW erwartungsgmäß ähnlich viel Energie, wie bei den

Tabelle 10.5: Vergleich der Leistungsaufnahme der drei Schaltungsvarianten

Bauteil	pwm-a	pwm-b(lang)	pwm-c(gigabit)
Pulstabelle	8,19 mW	3,52 mW	-
Pulsformer	2,64 mW	1,31 mW	5,20 mW
Tapped Delay-Line	2,77 mW	80,20 mW	5,46 mW
Einsenzähler	0,61 mW	1,21 mW	0,53 mW
Einsentabelle	14,76 mW	-	26,90 mW
PLL	77,51 mW	62,80 mW	45,47 mW
Nios II	176,99 mW	24,87 mW	27,16 mW
Top-Level-Entity	433,75 mW	203,73 mW	391,52 mW
Gesamt mit Peripherie	731,66 mW	416,86 mW	521,64 mW

anderen Schaltungsvarianten. Die übrigen Schaltungsteile benötigen ähnlich wenig Energie im einstelligen Milliwattbereich, wie bei Schaltungsvariane A. Die weitaus meiste Energie benötigt abermals die Top-Level-Entity mit 204 mW.

11 Diskussion

Die vorgestellten Ergebnisse zeigen, dass die asynchrone PWM-Datenübertragung höhere Datenraten erreichen kann, als die klassische Bit-serielle Übertragung, obwohl die Pulsweitenmodulation ebenfalls vollständig digital implementiert wird. Die Ergebnisse stellen Hochrechnungen anhand der verwendeten Schaltungsparameter dar. Leider kann die erreichbare Datenrate nicht im Dauerbetrieb gezeigt werden, da das Synthesewerkzeug sich bei der Umsetzung der asynchronen Schaltungsteile problematisch verhält. Nichtsdestotrotz zeigt die Arbeit auf den physikalischen Protokollschichten vielversprechende Ansätze für ein leistungsfähiges Kommunikationsverfahren.

Die Funktion der PWM-Datenübertragung erfordert eine Kalibrierung, um die Einflüsse der Übertragungsleitung und der Fertigungstoleranzen auszugleichen. Darüberhinaus ist eine regelmäßige Kalibrierung aufgrund von Temperaturänderungen notwendig. Diese Kalibrierung bringt jedoch auch einen Vorteil mit sich: Die PWM-Datenübertragung kann auf Übertragungsleitungen mit unbekannten elektrischen Eigenschaften eingesetzt werden. Im Rahmen der Kalibrierung werden dabei die Übertragungssymbole so festgelegt, dass die vorhandene Bandbreite ausgenutzt wird. Beispielsweise ist die Schaltungsvariante A unvorhergesehenerweise in der Lage, Daten über 74,4 m zu übertragen.

Datenrate

Die in den Ergebnissen vorgestellten Werte für Datenrate, Latenz und Energiebedarf sind Abschätzungen. Insbesondere berücksichtigen sie keine Übertragungsfehler, die bei Temperaturänderung, Jitter und Rauschen auftreten. Eine sinnvolle Berücksichtigung dieser Fehler setzt höhere Protokollebenen mit Mechanismen zur Vorwärtsfehlerkorrektur voraus, wie sie auch bei anderen Übertragungsverfahren eingesetzt werden. Dennoch zeigen die Ergebnisse gut, welchen Gewinn die PWM-Datenübertragung erreichen kann.

Zeitauflösung

Aus elektrischer Sicht ist die erreichte Zeitauflösung von 130 ps besonders auf der 223,2 m langen Übertragungsstrecke beeindruckend. Diese Zeitauflösung entspricht exakt der mittleren Signallaufzeit der verwendeten Verzögerungselemente.

Dass die Zeitquantisierung hier begrenzend für die Auflösung ist, bedeutet aber auch, dass bessere Schaltungsvarianten oder kleiner Strukturbreiten zu besseren Zeitauflösungen führen können. Beispielsweise sind mit aufwendigeren Zeitmessschaltungen (z.B. [Joo10; SJH15b]) die eine 30-fache Steigerung der Zeitauflösung möglich. Problematisch wirkt sich hierbei jedoch abermals der Nichtdeterminismus der FPGA-Synthese aus. Entsprechende Versuche, solche Zeitmessschaltungen für die PWM-Datenübertragung zu nutzen, blieben erfolglos.

Jitter

Die Präzision der Zeitmessung wird kurzfristig maßgeblich vom Jitter beeinflusst. Bei der Zeitauflösung von 130 ps waren jedoch nur kurzfristige Abweichungen in der letzten Bit-Stelle festzustellen. Die Ursachen können sehr vielfältig sein. So besagt die Bandbreite des 223,2 m-Kabels von 6,1 MHz, dass die 350 mV LVDS-Spannungshub innerhalb von 82,3 ns einmal umgeladen werden können. Die Flankensteilheit beträgt daher etwa

$$\frac{350\,\mathrm{mV}}{82{,}3\,\mathrm{ns}} = 4{,}25\,\mathrm{mV\,ns^{-1}}.$$

Innerhalb der 130 ps ändert sich die Spannung um $4{,}25\,\mathrm{mV\,ns^{-1}} \cdot 130\,\mathrm{ps} = 0{,}55\,\mathrm{mV}$. Dies deckt sich größenordnungsmäßig mit den am Oszilloskop ermittelten Signalformen (Abbildung 10.3). Daher ist es eher verwunderlich, dass der Empfänger nach 223,2 m die Pulse überhaupt noch unterscheiden kann. Warum die Pulsauflösung trotz aller Rauscheinflüsse und der geringen Bandbreite so gut ist, bleibt an dieser Stelle unklar.

Temperaturverhalten und weitere Fehlereinflüsse

Die Änderungen des Zeitverhaltens der asynchronen Schaltungsteile mit der Temperatur sind für die Datenübertragung signifikant. Allein Erwärmung des FPGA durch den Betrieb der Schaltung um etwa 30 K beeinflusst mehrere Bit-Stellen. Diese Problem wird durch Neukalibrieren bei Erreichen der Betriebstemperatur umgangen. Diese Lösung ist selbstverständlich nicht zufriedenstellend, da Temperaturänderungen im Betrieb durch äußere Einflüsse gänzlich vernachlässigt werden. Die Ergebnisse zeigen jedoch, dass alle Elemente einer Verzögerungskette relativ gleichmäßig von der Erwärmung betroffen sind. Zukünftig ließe sich beispielsweise auf einer höheren Protokollschicht jeder Datenwert als Differenz zum vorherigen Datenwert darstellen, wodurch ein konstanter Versatz aller Datenwerte ausgeglichen wird. Voraussetzung für diese sogenannte differentielle Codierung ist jedoch die Beherrschbarkeit des Syntheseprozesses.

Reproduzierbarkeit der Schaltungssynthese

Das Hauptproblem bei der Erstellung der Schaltung entsteht durch die Optimierungsversuche von Quartus. Das Synthesewerkzeug ist bemüht, das Zeitverhalten der synchronen Schaltungsteile günstig zu gestalten. Allerdings gibt es beispielsweise mit Reset-Leitungen auch Signale, die sowohl die synchronen, als auch die asynchronen Schaltungsteile erreichen. In der Folge haben Veränderungen in einem Schaltungsteil auch ein verändertes Zeitverhalten der anderen Schaltungsteile zur Folge. Das ist insbesondere dann ein Problem, wenn die asynchronen Schaltungsteile nach einer eigentlich unabhängigen Änderung nicht mehr funktionieren. Im Rahmen der Arbeit gelang es nicht, das Synthesewerkzeug in diesem Punkt soweit zu beherrschen, dass bei erfolgreicher Timing-Analyse auch immer eine funktionierende Schaltung herauskam. Daher mussten alle Syntheseergebnisse auf dem FPGA getestet werden. Der einzige Ausweg wäre, die Synthesewerkzeuge so anzupassen, dass das Zeitverhalten flexibler angegeben werden kann. Dann bräuchten asynchrone Schaltungsteile nicht mehr von der Analyse ausgenommen werden. Damit hätte das Synthesewerkzeug wieder einen globalen Überblick über das Zeitverhalten der Gesamtschaltung.

11.1 Ausblick

11.1.1 Fertigstellung Datenübertragung

Die vorliegende Arbeit beschreibt und implementiert die niedrigste Ebene der asynchronen PWM-Datenübertragung. Aus praktischer Sicht fehlen dem vorgestellten Übertragungssystem noch höhere Protokollschichten, die den Verbindungaufbau steuern. Weiterhin ist die Fehlerwahrscheinlichkeit der Datenübertragung aufgrund von Temperaturschwankungen relativ hoch. Eine entsprechende Vorwärtsfehlerkorrektur sollte im Protokoll vorgesehen werden. Eine Fehlererkennung kann auch für das Auslösen einer automatischen Neukalibrierung verwendet werden. Lässt die Anwendung eine Unterbrechung der Datenübertragung für Kalibrierungszwecke nicht zu, so ist eine Anpassung der Kalibrierdaten zur Laufzeit vorstellbar. Dabei kann die Eigenschaft der Pulsweitenmodulation ausgenutzt werden, dass weit auseinanderliegende Pulslängen auch bei großen Störungen noch am Empfänger unterscheidbar sind.

Verbindungsaufbau

Um die für den Verbindungsaufbau nötigen Informationen im Vorfeld zwischen den Kommunikationspartnern austauschen zu können, kann beispielsweise auf dem selben Kabel ein einfaches serielles Protokoll zum Einsatz kommen. Ist eine Datenverbindung per Pulsweitenmodulation erst einmal aufgebaut, ist es sinnvoll,

eine Vorwärtsfehlerkorrektur für die Erkennung von Übertragungsfehlern einzusetzen.

Erkennung von Übertragungsfehlern

Anzahl und Häufigkeit von Übertragungsfehlern geben Aufschluss über eine Abweichung der Schaltung von den Kalibrierdaten aufgrund von Temperaturänderungen. Die Erkennung der Fehler erfolgt immer beim Empfänger. Dieser muss daraufhin dem Sender einen Kalibrierbedarf signalisieren. Eine Möglichkeit besteht aus einem vollständigen Abbruch der Verbindung und anschließendem erneuten Verbindungsaufbau. Da dies für zeitkritische Anwendungen nicht immer tragbar ist, kann eine Kalibrierungsanpassung zur Laufzeit auch auf Basis der Art der erkannten Fehler geschehen. Sind beispielsweise alle Pulse zu kurz, werden vom Empfänger aber noch erkannt, so genügt es, die Symboltabelle des Empfängers zu modifizieren.

Austausch von Kalibrierdaten zu Laufzeit

Aus der Notwendigkeit des Austausches von Kalibrierdaten zur Laufzeit folgt unmittelbar der Bedarf nach einem In-Band-Signaling für diese Daten. Bei der Belegung der Übertragungssymbole führt das prinzipiell zu einer Reduzierung der nominellen Datenrate. Allerdings kann durch die Verringerung der Fehlerrate die effektive Datenrate erhöht werden. Die Parametrierung eines solchen Verfahrens muss aufgrund der Vielzahl an Einflussfaktoren praktisch erprobt werden.

Nichtganzzahlige Bit-Anzahl je Symbol

Eine ganzen Zahl von b Bit pro Puls erfordert eine Zweierpotenz 2^b verschiedener Symbole. Um die theoretisch mögliche Datenrate besser auszunutzen, ist es sinnvoll sämtliche zur Verfügung stehende Symbole zur Übertragung der Daten zu nutzen. Eine zusätzliche Verarbeitungsstufe in Sender und Empfänger kann diese Aufgabe übernehmen. Dabei kann eine Zustandsmaschine so implementiert werden, dass eine Symbolfolge einer bestimmten Bit-Kombination zugeordnet wird. Prinzipiell ist es dabei möglich, durch „halbe“ Bit die Datenrate zu erhöhen, und trotzdem eine sofortige Dekodierung der bereits vollständig übertragenen Bit durchzuführen.

Zur Illustration sei ein Empfänger angenommen, der sechs Symbole 0 bis 5 unterscheiden kann. Bei der Übertragung von 2 Bit pro Puls bleiben zwei Symbole ungenutzt. Betrachtet man eine Folge zweier Symbole, so lassen sich $6^2 = 36$ verschiedene Folgen unterscheiden. Durch die Zuordnung von 5 Datenbit zu den ersten 32 Symbolfolgen, lassen sich mit zwei Symbolen 5 Bit übertragen. Dieses

Schema lässt sich auch auf mehr als zwei Symbole anwenden. Bei einer Folge von drei Symbolen können sieben Bit übertragen werden. Im Grenzfall von unendlich vielen Symbolen lassen sich $log_2 6 \approx 2{,}58$ Bit pro Symbol übertragen. Für den Fall zweier Symbole werden 2,5 Bit pro Symbol codiert. Der Datenratengewinn natürlich von den genauen Schaltungsparamtern abhängig, kann aber im zweistelligen Prozentbereich liegen. Eine darüber hinausgehende Verwendung verbleibender Symbolkombinationen kann in der Signalisierung von Protokollinformationen liegen. Beispiele seien eine Neukalibrierung oder ein Nachrichtenstart.

11.1.2 Evaluation auf verschiedenen FPGAs

Reale Messung des Energiebedarfs

Die theoretische Betrachtung des Energiebedarfs ist mit großen Unsicherheiten verbunden. Eine Reihe von Einflussfaktoren ist nicht präzise genug festgelegt. Beispielsweise ist der Energiebedarf der externen Beschaltung nur experimentell bestimmbar. Eine Messung des Energiebedarfs ist ebenfalls notwendig, um die PWM-Datenübertragung mit anderen Übertragungssystemen auf der selben Plattform vergleichen zu können. Denn es ist zu erwarten, dass die PWM-Datenübertragung auf langen Leitungen, wegen der geringeren Signalflankenanzahl Vorteile hat.

Cross-Plattform Datenübertragung

Ein großer Vorteil der vorgestellten PWM-Datenübertragung ist die Implementierungsmöglichkeit auf verschiedenen Plattformen. Die logische Konsequenz ist, dies auch zu nutzen. Momentan ist das Anwendungsgebiet zwar relativ beschränkt, denn Schaltungen mit verschiedenen FPGAs womöglich auch noch von verschiedenen Herstellern sind selten. Aus Sicht der FPGA-Hersteller kann eine weitere schnelle Kommunikationsmöglichkeit über lange Kabel jedoch der Verbreitung von FPGAs insgesamt nur förderlich sein. Insbesondere die Implementierung in den Standardlogikelementen ermöglicht aus Kundensicht eine flexible Anpassung der Datenübertragungsstrecke an seine Erfordernisse, ohne dass der FPGA-Hersteller diese Anforderungen vorhersehen müsste.

11.1.3 Hardware-Optionen

LUTs als Symboltabelle

Da hohe Pulsraten durch den Durchsatz der RAM-basierten Symboltabellen begrenzt werden, wären alternative Symboltabellen hilfreich. Da bei hohen Pulsraten in der Regel nur 3 bis 5 Bit pro Puls übertragen werden, ist der Speicherbedarf für die Symboltabelle entsprechend gering. Somit könnten auch die LUTs als Speicher

genutzt werden. Bei dieser Implementierung gibt es nur ein großes Problem: Die Kalibrierdaten müssen im FPGA-Bit-Stream enthalten sein, da der Bit-Stream den Inhalt der LUT-Bits festlegt. Daher müsste nach jeder Kalibrierung der neue LUT-Inhalt an die richtige Position im FPGA geschrieben werden. Die größte Schwierigkeit ergibt sich aus der schnellen Generierung der neuen Bit-Stream, im Idealfall auch noch ohne angeschlossenen PC. Ein Unterfangen, das mit den heutigen Werkzeugen unmöglich ist. Auch deshalb sind weitere Forschungsarbeiten auf dem Gebiet der FPGA-Tools unumgänglich.

11.1.4 Beide Flanken modulieren

In der vorgestellten Arbeit hat ein PWM-Symbol unabhängig vom Datenwert immer die gleiche Länge T_{Symbol}. Eine Verdoppelung der Datenrate wäre möglich, wenn die Dauer des Low-Pegels ebenfalls anhand der Eingangsdaten moduliert würde. Diese Modulationsform wird auch als PIWM (Pulsintervall und -weitenmodulation) bezeichnet [Xio06, Seite 84]. Obwohl die Idee einfach zu beschreiben ist, wirft insbesondere die Senderimplementierung eine Reihe von Problemen auf. Die synchron ankommenden Daten auf eine asynchrone Schaltung mit variierender Datenrate umzusetzen ist nicht trivial. Die sich ergebende variable Pulsrate ist darüberhinaus nicht mit einem synchronen Zähler realisierbar, sodass weitere asynchrone Schaltungsteile benötigt werden. Das ist zwar nicht grundsätzlich schwierig, jedoch mit dem heutigen Stand der Synthesewerkzeuge nahezu unmöglich.

11.2 FPGA-Tools

Das dringlichste Problem bei der Implementierung der vorliegenden Schaltungen ist die Unfähigkeit der FPGA-Werkzeuge, die asynchronen Schaltungsteile zu verstehen. Die Entwicklung der Werkzeuge zur besseren Unterstützung asynchroner Schaltungen, ist für eine breite Verwendung unvermeidlich. Da dass Herstellerinteresse an solchen Nischenlösungen gering ist, ist die Entwicklung eigener Werkzeuge von Nöten. Dabei zeigt die historische Entwicklung, wie wichtig es ist, die Werkzeuge möglichst offen zu gestalten. Nur so können die dringend notwendigen neuen Ideen für FPGA-Anwendung überhaupt entwickelt und ausprobiert werden. Dass die FPGA-Community händeringend nach Anwendungsfeldern für FPGAs sucht, ist zu einem nicht unerheblichen Teil der schlechten Benutzbarkeit der FPGA-Werkzeuge zuzuschreiben. Da es sich dabei zum Großteil um herstellereigene Software handelt, ist eine schnelle Besserung ohne erkennbare Marktvorteile nicht zu erwarten. Einen möglichen Ausweg bieten Open-Source-Toolchains. Die bisher einzige ist „Project IceStorm“ [WL15], das von Wolf und Lasser für die iCE40-FPGA-Familie des Herstellers Lattice entwickelt wurde. Leider sind die

Hersteller in den letzten Jahren zunehmend restriktiver, was den Zugang zu Architekturinformationen der FPGAs angeht. Open-Source-Tools sind deshalb nur durch Reverse-Engineering erstellbar. Dies wird in der Regel von einigen wenigen Einzelpersonen durchgeführt. Dabei werden häufig nur genau diejenigen FPGAs und Funktionen unterstützt, die für den jeweiligen Entwickler von Bedeutung sind.

Literatur

[Agilent01] *Digital Modulation in Communications Systems - An Introduction*. Application Note 1298. Agilent Technologies, 14. März 2001. URL: http://cp.literature.agilent.com/litweb/pdf/5965-7160E.pdf (besucht am 29. 03. 2017) (siehe S. 27).

[Alo+07] Alberto Aloisio u. a. "FPGA Implementation of a High-Resolution Time-to-Digital Converter". In: Bd. 1. Okt. 2007, S. 504–507. DOI: 10.1109/NSSMIC.2007.4436379 (siehe S. 45).

[Bha99] Jayaram Bhasker. *A VHDL-Primer*. 3. Aufl. Upper Saddle River, NJ, USA: Prentice Hall PTR, 1999. ISBN: 0-13-096575-8 (siehe S. 33).

[CRM13] Daniel Costinett, Miguel Rodriguez und Dragan Maksimovic. "Simple Digital Pulse Width Modulator Under 100 ps Resolution Using General-Purpose FPGAs". In: *Power Electronics, IEEE Transactions on* 28.10 (Okt. 2013), S. 4466–4472. ISSN: 0885-8993. DOI: 10.1109/TPEL.2012.2233218 (siehe S. 58).

[CycloneII08] *Cyclone II Device Handbook*. Altera Document CII5V1-3.3. Altera Corporation. San Jose, CA, USA, 2008. URL: http://www.altera.com/literature/hb/cyc2/cyc2_cii5v1.pdf (besucht am 31. 12. 2016) (siehe S. 34, 37, 46, 84, 88, 95, 101).

[DH06] Ali Dasdan und Ivan Hom. "Handling Inverted Temperature Dependence in Static Timing Analysis". In: *ACM Transactions on Design Automation of Electronic Systems* 11.2 (2006). ISSN: 1084-4309. DOI: 10.1145/1142155.1142158. URL: http://ziyang.eecs.umich.edu/~dickrp/talp/papers/dasdan-temperature.pdf (besucht am 29. 03. 2017) (siehe S. 52).

[FC09] Claudio Favi und Edoardo Charbon. "A 17 ps Time-to-Digital Converter Implemented in 65 nm FPGA Technology." In: *FPGA*. Hrsg. von Paul Chow und Peter Y. K. Cheung. ACM, 24. Feb. 2009, S. 113–120. ISBN: 978-1-60558-410-2. URL: http://dblp.uni-trier.de/db/conf/fpga/fpga2009.html#FaviC09 (siehe S. 45, 50).

[FL12] Robert Frye und Kai Liu. “Meander Delay Compensation in High-Speed Digital Multilayer Packages”. In: *45th International Symposium on Microelectronics*. (San Diego, California, USA). 2012. DOI: 10.4071/isom-2012-TP27. URL: http://www.statschippac.com/services/documentlibrary/~/media/Files/DocLibrary/whitepapers/2012/STATSChipPAC_IMAPS_ISM2012_00034.ashx (besucht am 29. 03. 2017) (siehe S. 30).

[Flü11] Harald Flügel. *FPGA-Design mit Verilog*. Berlin, Boston: Oldenbourg Wissenschaftsverlag, 2011. ISBN: 3-486-71150-9 (siehe S. 33).

[HS16a] Matthias Hinkfoth und Ralf Salomon. “Calibration-Friendly FPGA-Based Time-to-Digital-Converter Element Combining Two Distinct Measurement Methods”. In: *Proceedings of the 12th IEEE International Conference on Control & Automation (IEEE ICCA)*. Kathmandu, Nepal, 2016, S. 179–185. ISBN: 978-1-5090-1738-6. DOI: 10.1109/ICCA.2016.7505272 (siehe S. 53).

[HS16b] Matthias Hinkfoth und Ralf Salomon. “Fast Bit Compressor: A Novel Implementation of a Low-Latency Modulation Using Binary Signals on Bandwidth-Limited Links”. In: 21st IEEE International Conference on Emerging Technologies and Factory Automation (ETFA). revised, accepted, presented, but not published due to neglection of IEEE’s copyright terms. Berlin, Germany, 2016. URL: http://www.amd.e-technik.uni-rostock.de/veroeff/ETFA2016_paper_135.pdf (besucht am 26. 03. 2017) (siehe S. 55).

[HS16c] Matthias Hinkfoth und Ralf Salomon. “Increasing the Utility of Self-Calibration Methods in High-Precision Time-Measurement Systems”. In: *Proceedings of the 23rd ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*. Abstract and Poster. New Nork, NY, USA: ACM, 2016, S. 275. ISBN: 978-1-4503-3856-1. DOI: 10.1145/2847263.2847311 (siehe S. 53).

[ISE1212] *Synthesis and Simulation Design Guide*. UG626. Xilinx Inc. 2100 Logic Drive, San Jose, CA, USA, 18. Dez. 2012. URL: http://www.xilinx.com/support/documentation/sw_manuals/xilinx14_7/sim.pdf (besucht am 14. 03. 2017) (siehe S. 33).

[JH15] Ralf Joost und Matthias Hinkfoth. “Combining BOUNCE and X-ORCA: Improving their Real-World Utility”. In: *Microprocessors and Microsystems* 39.8 (2015), S. 648–655. ISSN: 0141-9331. DOI: 10.1016/j.micpro.2015.08.009 (siehe S. 53).

[JHS13] Ralf Joost, Matthias Hinkfoth und Ralf Salomon. “Improving Calibration Precision of Signal-Delay-Based Time Measurement Systems in FPGAs”. In: *Reconfigurable Computing and FPGAs (ReConFig), 2013 International Conference on*. Dez. 2013, S. 1–6. ISBN: 978-1-4799-2079-2. DOI: 10.1109/ReConFig.2013.6732301 (siehe S. 53).

[Joo10] Ralf Joost. “BOUNCE: On-Chip Signalleitungen als Basis digitaler Zeitmessung”. Diss. Universität Rostock, 2010. URL: http://rosdok.uni-rostock.de/file/rosdok_derivate_4287/Dissertation_Joost_2010.pdf (besucht am 27. 03. 2017) (siehe S. 45, 53, 130).

[Kam08] Karl-Dirk Kammeyer. *Nachrichtenübertragung*. 4. Aufl. 2008. 844 S. ISBN: 3-8351-0179-X (siehe S. 29).

[KTR08] Ian Kuon, Russell Tessier und Jonathan Rose. *FPGA Architecture: Survey and Challenges*. Bd. 2. Foundations and Trends in Electronic Design Automation 2. Boston, MA, USA: now publishers inc., 2008, S. 135–253. ISBN: 978-1-60198-126-4. DOI: 10.1561/1000000005 (siehe S. 33).

[Mar+14] Jacques Marteau u. a. “Implementation of Sub-Nanosecond Time-to-Digital Convertor in Field-Programmable Gate Array: Applications to Time-of-Flight Analysis in Muon Radiography”. In: *Measurement Science and Technology* 25.3 (2014), S. 035101. ISSN: 0957-0233. DOI: 10.1088/0957-0233/25/3/035101. URL: http://iopscience.iop.org/article/10.1088/0957-0233/25/3/035101 (besucht am 29. 03. 2017) (siehe S. 45).

[Microlab13] *Line Stretchers, SR series*. Microlab. Parsippany, NJ, 23. Apr. 2013 (siehe S. 53).

[Ngu16] Jean-François Nguyen. *Analising the Bitstream of Altera's Max-V CPLDs*. 2016. URL: https://lse.epita.fr/lse-summer-week-2016/slides/lse-summer-week-2016-07-maxv_cpld.pdf (besucht am 24. 03. 2017) (siehe S. 89).

[Quartus13] *Quartus II Handbook*. Version 13.0. Altera Document QII5V1-13.0.0. Altera Corp. San Jose, CA, USA, Mai 2013. URL: http://wl.altera.com/literature/hb/qts/archives/quartusii_handbook_archive_130.pdf (besucht am 31. 12. 2016) (siehe S. 33, 40).

[RG316/U13] *RG 316 Datenblatt*. Haiba Cable. 12. Sep. 2013. URL: http://www.koax24.de/fileadmin/download/datasheet/de/050327_Datenblatt_30000-316-50_RG316_C.pdf (besucht am 29. 03. 2017) (siehe S. 17).

[Rud05] Prof. Dr.-Ing. Dietmar Rudolph. *Tiefpaß-Systeme*. 2005. URL: http://diru-beze.de/signale/skripte/index.html (siehe S. 15).

[Sch80] Bernd Schott. "Datenübertragungssystem zwischen wenigstens zwei Mikroprozessorsystemen". 3035804A1 (DE). Robert Bosch GmbH. Sep. 1980 (siehe S. 29, 30).

[Sha49] Claude E. Shannon. "Communication in the Presence of Noise". In: *Proceedings of the IRE*. Bd. 37. 1. (Der Nachdruck von 1998 ist fehlerhaft.) 1949, S. 10–21. DOI: 10.1109/JRPROC.1949.232969. URL: https://web.stanford.edu/class/ee104/shannonpaper.pdf (besucht am 29. 03. 2017) (siehe S. 15, 18).

[SHJ12] Ralf Salomon, Enrico Heinrich und Ralf Joost. "Modeling the Nucleus Laminaris of the Barn Owl: Achieving 20 ps Resolution on a 85MHz-Clocked Digital Device". In: *Frontiers in Computational Neuroscience* 6.6 (2012). ISSN: 1662-5188. DOI: 10.3389/fncom.2012.00006. URL: http://www.frontiersin.org/computational_neuroscience/10.3389/fncom.2012.00006/abstract (siehe S. 45).

[Siemon97] *Propagation Delay and Delay Skew*. Watertown, CT, USA: The Siemon Company, 3. Juni 1997. URL: http://www.siemon.com/us/white_papers/97-06-03-delayskew.asp (besucht am 28. 12. 2016) (siehe S. 18).

[SJH15a] Ralf Salomon, Ralf Joost und Matthias Hinkfoth. "Platform-Independent Gigabit Communication for Low-Cost FPGAs". In: *Proceedings of the 2015 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*. Abstract and Poster. New York, NY, USA: ACM, 2015, S. 265. ISBN: 978-1-4503-3315-3. DOI: 10.1145/2684746.2689150 (siehe S. 55).

[SJH15b] Ralf Salomon, Ralf Joost und Matthias Hinkfoth. "Verfahren und System zur Übertragung von Daten". 10 2014 201 807.2 (DE). Universität Rostock. 26. Feb. 2015. URL: http://www.tbi-mv.de/fileadmin/user_upload/patente/archiv/DE102014201807B3.pdf (besucht am 26. 03. 2017) (siehe S. 55, 130).

[SK10] R. Szplet und K. Klepacki. "An FPGA-Integrated Time-to-Digital Converter Based on Two-Stage Pulse Shrinking". In: *Instrumentation and Measurement, IEEE Transactions on* 59.6 (Juni 2010), S. 1663–1670. ISSN: 0018-9456. DOI: 10.1109/TIM.2009.2027777 (siehe S. 45).

[SKJ09] R. Szplet, J. Kalisz und Z. Jachna. "A 45 ps time digitizer with a two-phase clock and dual-edge two-stage interpolation in a field programmable gate array device". In: *Measurement Science and*

Technology 20 (Jan. 2009). DOI: 10.1088/0957-0233/20/2/025108 (siehe S. 52).

[Spartan6] *Spartan-6 Family Overview*. URL: http://www.xilinx.com/support/documentation/data_sheets/ds160.pdf (besucht am 23. 03. 2017) (siehe S. 98).

[Szp+13] R Szplet u. a. "A 2.9 ps Equivalent Resolution Interpolating Time Counter Based on Multiple Independent Coding Lines". In: *Measurement Science and Technology* 24.3 (2013), S. 035904. ISSN: 0957-0233. DOI: 10.1088/0957-0233/24/3/035904 (siehe S. 45, 49, 98).

[TI02] *LVDS Application and Data Handbook*. Texas Instruments, Nov. 2002. URL: http://www.ti.com/lit/ug/slld009/slld009.pdf (besucht am 29. 12. 2016) (siehe S. 21).

[TIA16] *Commersal Building Telecommunications Cabling Standard Set*. Preis: 1615 US-$. Telecommunications Industry Association (TIA), 2016. URL: https://global.ihs.com/doc_detail.cfm?item_s_key=00378460 (besucht am 14. 03. 2017) (siehe S. 113, 114).

[Timequest09] *SDC and TimeQuest API Reference Manual*. Version 5.0. Altera Document MNL-SDCTMQ-5.0. Altera Corp. San Jose, CA, 2009. URL: http://www.altera.com/literature/manual/mnl_sdctmq.pdf (besucht am 06. 01. 2017) (siehe S. 80).

[usb-loss15] *USB 3.1 Loss Budget*. 2. März 2015. URL: http://www.usb.org/developers/docs/whitepapers/USB_3.1_Loss_Budget_Rev_1.0_-_2015-03-02.pdf (besucht am 12. 02. 2017) (siehe S. 19).

[usb-spec13] *Universal Serial Bus 3.1 Specification*. USB_3_1_r1.0.pdf aus [usb17]. 26. Juli 2013 (siehe S. 14, 21, 22).

[usb-typec16] *USB Type C Specification*. USB Type C/USB Type C Specification Relase 1.2.pdf aus [usb17]. 25. März 2016 (siehe S. 19).

[usb17] *USB 3.1 Specification*. 13. Jan. 2017. URL: http://www.usb.org/developers/docs/usb_31_011317.zip (besucht am 13. 02. 2017) (siehe S. 141).

[Wan+10] Jinhong Wang u. a. "A Fully Fledged TDC Implemented in Field-Programmable Gate Arrays". In: *Nuclear Science, IEEE Transactions on* 57.2 (Apr. 2010), S. 446–450. ISSN: 0018-9499. DOI: 10.1109/TNS.2009.2037958 (siehe S. 45).

[Wan+11] Jinhong Wang u. a. "The 10-ps Multitime Measurements Averaging TDC Implemented in an FPGA". In: *Nuclear Science, IEEE Transactions on* 58.4 (Aug. 2011), S. 2011–2018. ISSN: 0018-9499. DOI: 10.1109/TNS.2011.2158551 (siehe S. 45).

[WL15] Clifford Wolf und Mathias Lasser. *Project IceStorm*. 2015. URL: `http://www.clifford.at/icestorm/` (besucht am 13. 01. 2017) (siehe S. 134).

[WS08] Jinyuan Wu und Zonghan Shi. "The 10-ps Wave Union TDC: Improving FPGA TDC Resolution Beyond its Cell Delay". In: *Nuclear Science Symposium Conference Record, 2008. NSS '08. IEEE*. Okt. 2008, S. 3440–3446. DOI: `10.1109/NSSMIC.2008.4775079` (siehe S. 54, 78, 119).

[Wu09] Jinyuan Wu. "An FPGA Wave Union TDC for Time-of-Flight Applications". In: *Proceedings, 2009 IEEE Nuclear Science Symposium and Medical Imaging Conference (NSS/MIC 2009)*. 2009, S. 299–304. DOI: `10.1109/NSSMIC.2009.5401738`. URL: `http://lss.fnal.gov/cgi-bin/find_paper.pl?conf-09-573` (siehe S. 45).

[Xio06] Fuqin Xiong. *Digital Modulation Techniques*. 2. Aufl. 2006. 1017 S. ISBN: 1-58053-863-0 (siehe S. 13, 28, 134).

Thesen

1. Die Pulsweitenmodulation ist geeignet Daten zu übertragen. Dabei codiert die Pulslänge den Datenwert. Dank einer vollständig digitalen Implementierung weist die PWM-Datenübertragung nur geringe Hardware-Anforderungen auf.
2. Abhängig von der Zeitauflösung von Modulator und Demodulator, und anhängig von der zur Verfügung stehenden Bandbreite, kann die PWM-Datenübertragung eine spektrale Effizienz von mehr als 6 Bit/s/Hz erreichen.
3. Die maximal mögliche spektrale Effizienz der Bit-seriellen Übertragung von 2 Bit/s/Hz wird durch die PWM-Datenübertragung deutlich übertroffen.
4. Durch die Auswahl der zu übertragenden Pulse kann die PWM-Datenübertragung im Betrieb adaptiv auf Bandbreiteanforderungen eingestellt werden. Diese Adaptivität ist vorteilhaft, wenn die exakte Kabelbandbreite unbekannt ist.
5. Die Nutzung asynchroner Zeitmessschaltungen erhöht die Zeitauflösung der PWM-Datenübertragung um mehr als eine Größenordnung.
6. Mittels asynchroner Schaltungen lässt sich die Performance der PWM-Datenübertragung bis in den Bereich von Gigabit pro Sekunde steigern.
7. Die asynchrone PWM-Datenübertragung stellt geringe Anforderungen an die Taktfrequenz der synchronen Schaltungsteile. Es ist ausreichend, eine Taktfrequenz in der Größe der Pulsrate zu wählen.
8. Die Implementierung asynchroner Schaltungen in FPGAs ist ohne explizite Platzierung der Schaltungselemente möglich. Damit wird eine erhöhte Portabilität zwischen verschiedenen FPGAs der gleichen FPGA-Familie erreicht.
9. Die aktuelle Ausrichtung der FPGA-Entwurfswerkzeuge auf synchrone Schaltkreise erschwert die Erstellung asynchroner Strukturen.
10. Der Mangel an Dokumentation über den internen Aufbau von FPGAs verhindert weitestgehend alternative Wege in der FPGA-Synthese. Einzig der FPGA-Hersteller verfügt über das vollständige Wissen, um Synthesewerkzeuge zu erstellen. Dieses Wissen wird als Geschäftsgeheimnis gehütet.
11. Eine flexible, anpassbare Synthese-Software ist unabdingbar für eine reproduzierbare Entwicklung neuartiger Ideen mit FPGAs.

12. Die PWM-Datenübertragung ist nicht auf elektrische Signale beschränkt. Jedes binäre Signal ist für die Pulsweitenmodulation nutzbar. So ist beispielsweise die Nutzung optischer Übertragungsmedien denkbar.

Kurzreferat

Die Datenübertragung zwischen elektronischen Geräten ist seit Jahrzehnten ein aktuelles Forschungsgebiet. Ein prominentes Beispiel ist der Versuch, dem heimischen DSL-Anschluss immer höhere Datenraten zu entlocken. Das grundsätzliche Problem dabei ist die begrenzte Bandbreite des Kabels. Vor dem Hintergrund des steigenden Datenratenbedarfs versprechen höherwertige Modulationsverfahren Abhilfe, benötigen aber analoge Hardware für die Modulation. Im Vergleich zu digitalen Schaltungen erfordern die notwendigen analogen Schaltkreise lange Entwicklungszeiten. Platz- und Energiebedarf sind ebenfalls größer. Darüber hinaus erhöhen zusätzliche Verarbeitungsstufen die Übertragungsdauer für kleine Datenmengen.

Zur Vermeidung der Nachteile analoger Schaltungsteile stellt die vorliegende Arbeit eine neue Architektur für ein Datenübertragungssystem mittels Pulsweitenmodulation vor. Dabei wird der übertragene Datenwert durch eine Pulslänge repräsentiert. Wesentlicher Vorteil der Pulsweitenmodulation gegenüber anderen Modulationen ist der Verzicht auf analoge Schaltungen. Daraus ergeben sich ein geringerer Schaltungsaufwand und ein geringerer Energiebedarf. Im Gegensatz zu bisherigen Implementierungen setzt diese Arbeit asynchron arbeitende Modulations-Hardware für die PWM-Datenübertragung ein. Dadurch erreicht die vorgestellte Hardware, trotz vollständig digitaler Umsetzung, höhere Datenraten als Bit-serielle Datenübertragungen. Aufbauend auf einer theoretischen Betrachtung der PWM-Datenübertragung wird ein Kalibrierverfahren entwickelt, das die Anpassung der Modulationsparameter an die Bandbreite des Übertragungskabels ermöglicht. So kann durch Symbolauswahl die zur Verfügung stehende Kanalbandbreite ausgenutzt werden, ohne dass diese im Vorfeld bekannt sein muss.

Die Arbeit zeigt theoretisch und praktisch, dass eine Erhöhung der Zeitauflösung der Endgeräte zu einer höheren Datenrate führt. Die Umsetzung der Pulsweitenmodulation als Labormuster findet auf DE2-70 Entwicklungsplatinen mit Cyclone II FPGAs statt. Als Zeitmesssystem dient eine Tapped Delay-Line, wodurch sich die Zeitauflösung der PWM von 2,5 ns um mehr als eine Größenordnung auf 130 ps verbessert. Verschiedene Implementierungen werden mit Kabellängen von 1 m bis 223,2 m vermessen. Die Ergebnisse zeigen, dass die PWM-Datenübertragung durchweg höhere Datenraten erzielt als eine Bit-serielle Übertragung auf dem verwendeten FPGA. Bei 223,2 m wird die Datenrate um den Faktor 3,3 auf 40 MBit/s gesteigert.

Abstract

Data Transmission between electronic devices is an important research topic. A well known example is the continuous development of new DSL standards, each of which further boosts the data rate. The fundamental problem lies in the limited bandwidth of the cable. Therefore, sophisticated modulation schemes are required, in order to increase the date rate. But those modulation relies on special analog or mixed circuits, which require enormous development effort. Also, the additional energy consumption cannot be neglected. Finally, the additional processing stages add additional latency to the data transmission.

In oder to overcome the problems that arise from the utilization of analog circuits, this work proposes an entirely digital data transmission based on pulse width modulation (PWM). With PWM, the pulse width represents the actual data value. A novel implementation, based on asynchronous logic leads to impressive data rates, which outperform traditional bit-serial data transmission. A mathematical calculation of the properties of the PWM data transmission leads to the development of a calibration process, in order to adapt the modulation to the actual bandwidth of the cable. Moreover, this work shows that enhancements in the timing resolution of both endpoint devices directly lead to a higher data rate.

The PWM data transmission principle is applied to DE2-70 development boards hosting Cyclone II FPGAs. The prototype comprises a tapped delay-line for time measurement, thereby achieving a timing resolution of 130 ps. Several cables with lengths from 1 m to 223,2 m are tested. The results show, that the PWM data transmission can achieve better performance than a bit-serial transmission using this FPGA-type. With the 223,2 m cable, the data rate is increased by a factor of 3,3 , resulting in 40 MBit/s.

Danksagung

Danke Ralf Salomon, für die interessanten und lehrreichen Jahre in deiner Arbeitsgruppe. Deine persönliche und fachliche Unterstützung haben mir bei der Erstellung der Arbeit sehr geholfen. Und auch danke dafür, dass du im rechten Moment auch mal Chef warst. ;-)

Danke Ralf Joost, für die vielen Diskussionen, Fragen und Antworten, die meine Arbeit bereichert haben. Danke, dass du mir auch immer wieder alternative Sichtweisen eröffnet hast.

Weiterhin danke ich allen Mitarbeitern der Arbeitsgruppe, insbesondere auch den ehemaligen Mitarbeitern und den Studenten. Danke für die offene, häufig angenehm pragmatische und lehrreiche Zusammenarbeit.

Darüber hinaus danke ich den Mitarbeitern des Institut für Angewandte Mikroelektronik und Datentechnik für das angenehme Miteinander und die anhaltende Unterstützung.

Schließlich danke ich meiner Familie, die mich stets unterstützt hat. Besonderer Dank gilt dabei meiner Frau, die es in einer organisatorischen Glanzleistung vollbracht hat, neben dem Haushalt, den Kindern und allem anderen auch noch Korrektur zu lesen, sodass ich mich auf das Schreiben konzentrieren konnte. Danke Conny, ohne dich würde ich wahrscheinlich immer noch schreiben.